



Acrobat Core API Reference

Adobe Developer Relations

Technical Note #5191



Revised: March 10, 1999

Copyright 1995-1999 Adobe Systems Incorporated. All rights reserved.

NOTICE: All information contained herein is the property of Adobe Systems Incorporated.

No part of this publication (whether in hardcopy or electronic form) may be reproduced or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written consent of the publisher.

PostScript is a trademark of Adobe Systems Incorporated. All instances of the name PostScript in the text are references to the PostScript language as defined by Adobe Systems Incorporated unless otherwise stated. The name PostScript also is used as a product trademark for Adobe Systems' implementation of the PostScript language interpreter.

Except as otherwise stated, any reference to a "PostScript printing device," "PostScript display device," or similar item refers to a printing device, display device or item (respectively) that contains PostScript technology created or licensed by Adobe Systems Incorporated and not to devices or items that purport to be merely compatible with the PostScript language.

Adobe, the Adobe Logo, Acrobat, the Acrobat Logo, Acrobat Capture, Acrobat Exchange, Distiller, PostScript, and the PostScript logo are trademarks of Adobe Systems Incorporated.

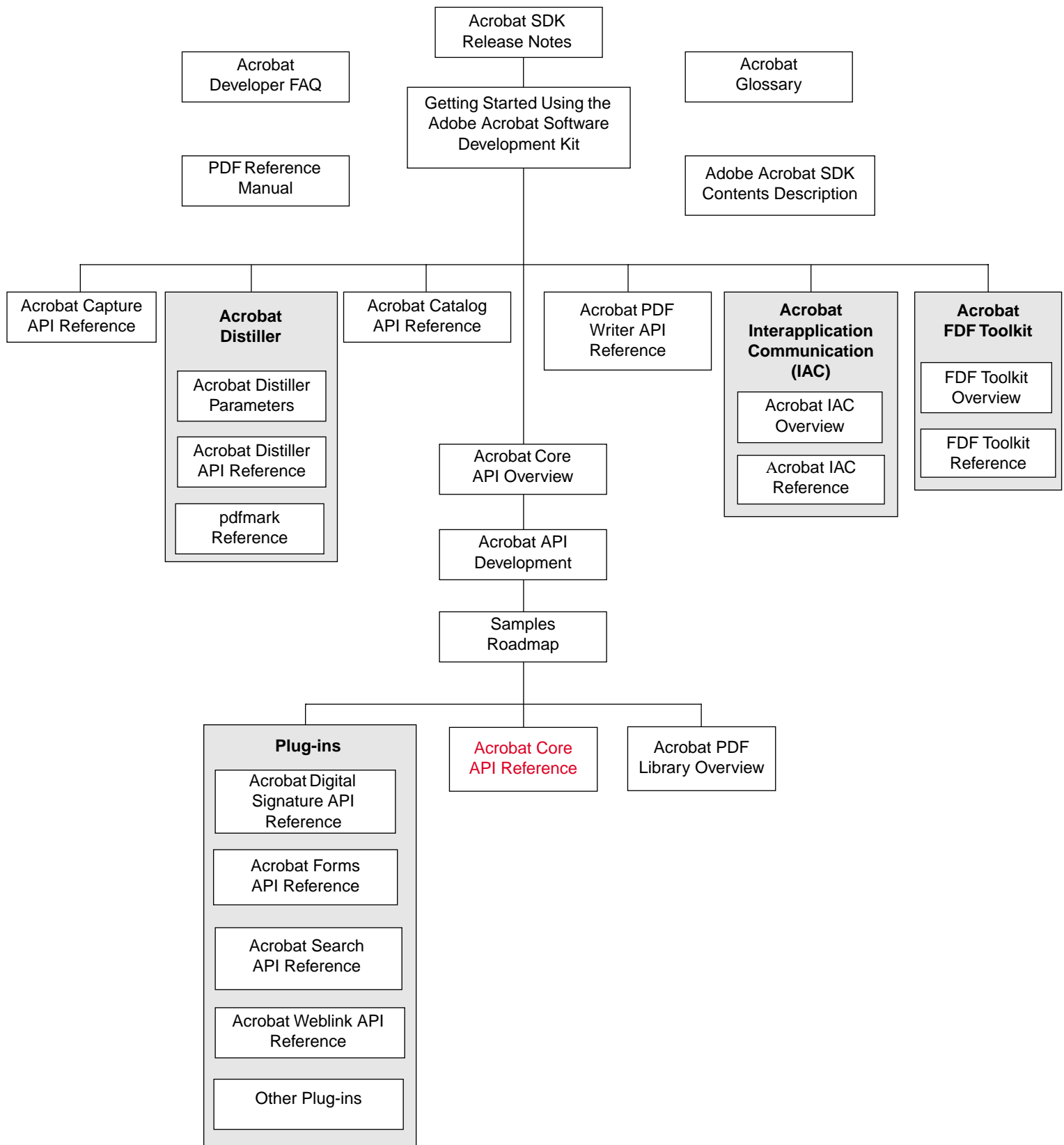
Apple, Macintosh, and Power Macintosh are trademarks of Apple Computer, Inc., registered in the United States and other countries. HP-UX is a registered trademark of Hewlett-Packard Company. AIX and PowerPC are registered trademarks of IBM Corporation in the United States. ActiveX, Microsoft, Windows, and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and other countries. UNIX is a registered trademark of The Open Group. All other trademarks are the property of their respective owners.

This publication and the information herein is furnished AS IS, is subject to change without notice, and should not be construed as a commitment by Adobe Systems Incorporated. Adobe Systems Incorporated assumes no responsibility or liability for any errors or inaccuracies, makes no warranty of any kind (express, implied, or statutory) with respect to this publication, and expressly disclaims any and all warranties of merchantability, fitness for particular purposes, and noninfringement of third party rights.

| Version History | | |
|-------------------|--------------------------|---|
| 15 September 1995 | Tim Bienz | First 2.1 version; split between Overview, Online Reference, and Development. |
| 25 January 1996 | Tim Bienz | Updated 2.1 version. |
| 30 November 1996 | Gary Staas, Tim Bienz | First 3.0 version. |
| 14 February 1997 | Gary Staas | Minor corrections and updates. |
| 5 June 1997 | Gary Staas | 3.0J and 3.01 version. |
| 23 June 1997 | Gary Staas | Minor revisions. |
| 23 September 1997 | Gary Staas | Minor revisions. |

| Version History | | |
|------------------|--------------|---|
| 14 May 1998 | Gary Staas | Added PDFEdit information, Adobe PDF Library information, changed title, added miscellaneous information. Changed Tech Note #. |
| 8 September 1998 | Denise Stone | Updated 4.0 version. |
| 30 October 1998 | Denise Stone | Minor revisions. |
| 26 November 1998 | Denise Stone | 4.0 revisions. |
| 14 December 1998 | Denise Stone | 4.0 revisions. |
| 1 February 1999 | Gary Staas | Minor revisions. |

Documentation Roadmap



Contents

- ◆ How to Use the Core API Reference 6
- ◆ Methods 10
- ◆ Callbacks 1628
- ◆ Declarations 1897
- ◆ Lists 2135
- ◆ Notifications 2167
- ◆ Errors 2258
- ◆ Macros 2592
- ◆ Objects 2638
- ◆ API Changes 2753

Using This Reference

How to Use the Core API Reference

Using This Reference

Description

This Core API technical note is intended for experienced users of the Adobe Acrobat Software Development Kit (SDK). It describes in detail the objects and methods provided by the Acrobat Application Program Interface (API).

In addition to having a working knowledge of ANSI C programming, it is useful to have a good understanding of object-oriented programming concepts, such as the use of methods, classes, objects, and member functions (callbacks).

Contents

This technical note contains the following sections:

- [Methods](#) descriptions. Detailed description of each method, including its parameters and return value.
- [Callbacks](#) descriptions. Detailed description of callback functions, in which the Acrobat viewer and Library call back plug-ins and applications.
- [Declarations](#) descriptions. Detailed descriptions of data structures used by the Acrobat viewer and Library.
- [Lists](#). List of useful information, such as UI element names.
- [Notifications](#) descriptions. Description of objects that allow a plug-in or application to indicate an interest in a specified event, and provide a procedure that is called each time that event occurs.
- [Errors](#) descriptions. List of error categories and error codes. These error codes typically represent exceptions raised by Acrobat.
- [Macros](#) descriptions. Detailed description of each macro, which includes operators on fixed-point numbers, conversions between integers and fixed-point numbers, conversions between C strings and fixed-point numbers, math and rectangle utilities, and matrix operations.
- [Objects](#). Descriptions of the objects, list of the methods that obtain and dispose of the objects, their accessor functions, Cos conversion, and validity testing.
- [API Changes](#). Description of new objects, methods, callbacks, and so forth, that have been added or changed extensively in Acrobat 4.0.

The color codes shown above are used throughout the document to help you identify the different types of objects. Words that are underlined are links that take you to a detailed discussion of that object.

Using This Reference

Where appropriate, sections are sub-divided into the following layers or models:

- Acrobat Support (AS) layer, which provides a variety of utility methods, including platform-independent memory allocation and fixed-point math utilities.
- Acrobat Viewer (AV) layer, which deals with the viewer's user interface.
- Cos Object System (Cos) layer, which provides access to the low-level building blocks of PDF files.
- Portable Document (PD) layer, which provides access to components of PDF documents.
- PDFEdit, which provides access to PDF page counts associated with the Contents key in the page's directory.
- PDSEdit, which provides access to the logical structure of a PDF document.

Other Useful Documentation

Readers are assumed to be familiar with the Acrobat API. The following technical notes provide this information.

[*Getting Started Using the Adobe Acrobat Software Development Kit \(SDK\)*](#), introduces the SDK, plug-ins, Interapplication Communication (IAC), libraries, toolkits, and Host Function Tables (HFTs). In addition to SDK installation instructions, this manual has an extensive "road map" to other documentation.

[*Acrobat Core API Overview*](#), Technical Note #5190. Gives an overview of the objects and methods provided by the Acrobat Core API.

[*Portable Document Format Reference Manual*](#), Version 1.3, provides a description of the PDF file format, as well as suggestions for producing efficient PDF files. It is intended for application developers who wish to produce PDF files directly.

Using This Reference

Conventions Used in This Book

Text styles are used to identify various operators, keywords, terms, and objects. Four formatting styles are used in this book:

- PostScript language operators, PDF operators, PDF keywords, the names of keys in dictionaries, and other predefined names are written in boldface. Examples are **moveto**, **Tf**, **stream**, **Type**, and **MacRomanEncoding**.
- Operands of PDF operators are written in an italic san serif font. An example is *linewidth*.
- Object types are written with initial capital letters and italics. An example is *FontDescriptor*.
- The first occurrence of terms and the Boolean values *true* and *false* are written in italics. This style is also used for emphasis.

Methods

AS Layer

ASAtom

ASAtomExistsForString

```
ASBool ASAtomExistsForString (const char* nameStr,
    ASAtom* atom);
```

| | |
|------------------------|---|
| Description | Tests whether or not an <i>ASAtom</i> exists for the specified string. |
| Parameters | <p><i>nameStr</i></p> <p>The string to test.</p> <p><i>atom</i></p> <p>(Filled by the method) If the <i>ASAtom</i> corresponding to <i>nameStr</i> already exists, it is returned in <i>atom</i>. Pass <i>NULL</i> for <i>atom</i> to simply check whether or not the <i>ASAtom</i> already exists, but do not get it if it does.</p> |
| Return Value | <i>true</i> if an <i>ASAtom</i> already exists for <i>nameStr</i> , <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CorCalls.h</i> |
| Related Methods | ASAtomFromString ASAtomGetCount |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASAtomFromString

[ASAtom](#) ASAtomFromString (const char* nameStr);

Description

Gets the *ASAtom* for the specified string. You can also use this method to create an *ASAtom*, since it creates one for the string if one does not already exist.

If an *ASAtom* already exists for *nameStr*, the existing *ASAtom* is returned. Thus *ASAtoms* may be compared for equality.

Because *ASAtoms* cannot be deleted, they are useful for strings that are used many times in an Acrobat viewer session, but are not advisable for strings that have a short lifetime. For the same reason, it is not a good idea to create large numbers of *ASAtoms*.

Parameters

nameStr

The string for which an *ASAtom* is created.

Return Value

The *ASAtom* corresponding to *nameStr*.

Exceptions

None

Notifications

None

Header File

CorCalls.h

Related Methods

[ASAtomExistsForString](#)
[ASAtomGetCount](#)

Example

```
/* Compare ASAtoms for equality */
PDAnnot annot;

if (PDAnnotGetSubtype(annot) ==
    ASAtomFromString("Link")) {
    ...
}
```

Availability

Plug-ins: Available if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASAtomGetString

```
const char* ASAtomGetString (ASAtom atom);
```

| | |
|------------------------|---|
| Description | Gets the string associated with the specified <i>ASAtom</i> . |
| Parameters | <p><i>atom</i></p> <p>The <i>ASAtom</i> whose string is obtained.</p> |
| Return Value | The string corresponding to <i>atom</i> . Returns <i>NULL</i> or an empty string if the <i>atom</i> has not been defined. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CorCalls.h</i> |
| Related Methods | ASAtomExistsForString ASAtomFromString |
| Availability | Plug-ins: Available if <i>PL_ACROSSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASCallback

ASCallbackCreate

[ASCallback](#) `ASCallbackCreate (ASExtension extensionID, void* proc);`

Description

Creates a callback that allows the Acrobat viewer to call a function in a plug-in. All plug-in functions that are called by the Acrobat viewer must be converted to callbacks before being passed to the viewer.

Whenever possible, plug-ins should not call `ASCallbackCreate` directly, but should use the macros [ASCallbackCreateProto](#), [ASCallbackCreateNotification](#), and [ASCallbackCreateReplacement](#). These macros (which eventually call `ASCallbackCreate`) have two advantages:

- They allow compilers to perform type checking, eliminating one extremely common source of plug-in bugs.
- They handle *extensionID* automatically.

Plug-ins must use `ASCallbackCreate` directly, for example, when calling a Macintosh toolbox routine that expects a *ProcPtr*.

Parameters

extensionID

The [gExtensionID](#) extension that calls *proc*.

proc

The user-supplied procedure for which a callback is created.

Return Value

The newly-created callback.

Exceptions

None

Notifications

None

Header File

CorCalls.h

Related Methods

[ASCallbackDestroy](#)

Related Macros

[ASCallbackCreateNotification](#)
[ASCallbackCreateReplacement](#)
[ASCallbackCreateProto](#)

Availability

Plug-ins: Available if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASCallbackDestroy

```
void ASCallbackDestroy (ASCallback callback);
```

| | |
|------------------------|--|
| Description | Destroys a callback. |
| Parameters | <p><i>callback</i></p> <p>The callback to destroy.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CorCalls.h</i> |
| Related Methods | ASCallbackCreate |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASExtension

ASEnumExtensions

```
ASExtension ASEnumExtensions (ASExtensionEnumProc proc,  
void* clientData, ASBool onlyLivingExtensions);
```

| | |
|------------------------|--|
| Description | Enumerates all <i>ASExtensions</i> . |
| Parameters | <p><i>proc</i></p> <p>User-supplied callback to call for each plug-in. Enumeration halts if <i>proc</i> returns <i>false</i>.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>proc</i> each time it is called.</p> <p><i>onlyLivingExtensions</i></p> <p>If <i>true</i>, <i>ASExtensions</i> that have been unloaded or otherwise deactivated are not enumerated. If <i>false</i>, all <i>ASExtensions</i> are enumerated.</p> |
| Return Value | If <i>proc</i> returned <i>false</i> , the last <i>ASExtension</i> that was enumerated. <i>NULL</i> otherwise. |
| Notifications | None |
| Header File | <i>CorCalls.h</i> |
| Related Methods | ASExtensionGetFileName ASExtensionGetRegisteredName |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

ASExtensionGetFileName

```
ASInt32 ASExtensionGetFileName (ASExtension extension,
    char* buffer, ASInt32 bufSize);
```

Description Derives a readable name for the *ASExtension* from the handler data only.

Parameters

extension

The *ASExtension* whose file name is obtained.

buffer

(Filled by the method) Pointer to a buffer for the file name.

bufSize

(Filled by the method) Number of bytes in *buffer*.

Return Value Number of bytes in the file name.

Notifications None

Header File *CorCalls.h*

Related Methods [ASExtensionGetRegisteredName](#)

Availability Plug-ins: Available if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

ASExtensionGetRegisteredName

```
ASAtom ASExtensionGetRegisteredName (ASExtension extension);
```

| | |
|------------------------|--|
| Description | Gets the registered name associated with a plug-in. |
| Parameters | <i>extension</i> The <i>ASExtension</i> whose name is obtained. |
| Return Value | An <i>ASAtom</i> representing the plug-in name. |
| Notifications | None |
| Header File | <i>CorCalls.h</i> |
| Related Methods | ASExtensionGetFileName |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

ASFile

ASFileAcquirePathName

```
ASPathName ASFileAcquirePathName (ASFile aFile);
```

Description

Gets the pathname for *aFile* and increments an internal reference count. When you are done using the pathname, call [ASFileSysReleasePath](#) to decrement the reference count. The format of the returned pathname is platform-specific.

Note: The [ASPathName](#) returned should be released with the [ASFileSysReleasePath](#) method when it is no longer needed.

Parameters

aFile

The file whose pathname is acquired.

Return Value

The file's *ASPathName*. You can use [ASFileSysDIPathFromPath](#) to convert this to a device-independent pathname.

After you are done using the *ASPathName*, you must free it using [ASFileSysReleasePath](#).

Exceptions

None

Notifications

None

Header File

ASCalls.h

Related Methods

[ASFileSysReleasePath](#)

Availability

Plug-ins: Available if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileClose

```
ASInt32 ASFileClose (ASFile aFile);
```

| | |
|------------------------|--|
| Description | Closes the specified file. After a call to <i>ASFileClose</i> , the file is no longer valid and may be reused as the result of a subsequent call to ASFileSysOpenFile . |
| Parameters | <p><i>aFile</i></p> <p>The open file to associate with the stream. The file must have been opened previously using ASFileSysOpenFile.</p> <p>Each open file has an unique <i>ASFile</i>.</p> |
| Return Value | Error if <i>aFile</i> is not open. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASMemStmRdOpen ASFileStmRdOpen ASProcStmRdOpen |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileFlush

```
void ASFileFlush (ASFile aFile);
```

| | |
|------------------------|--|
| Description | Flushes any buffered data to a file. |
| Parameters | <i>aFile</i> The file whose data is flushed. |
| Return Value | None |
| Exceptions | fileErrIO |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileFromMDFile

```
ASBool ASFileFromMDFile (MDFile fileID, ASFileSys fileSys,
    ASFile* pfN);
```

| | |
|------------------------|--|
| Description | Gets the <i>ASFile</i> associated with the specified <i>MDFile</i> and <i>ASFileSys</i> . |
| Parameters | <p><i>fileID</i></p> <p>The <i>MDFile</i> for which the information is desired.</p> <p><i>fileSys</i></p> <p>The <i>ASFile</i> for which the information is desired.</p> <p><i>pfN</i></p> <p>(Filled by the method) Return pointer to the <i>ASFile</i> for <i>fileID</i> and <i>fileSys</i>.</p> |
| Return Value | <i>true</i> if <i>fileID</i> is an open file, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileGetFileSys ASFileGetMDFile |
| Availability | Plug-ins: Available if <i>PI_ACROSSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileGetEOF

```
ASInt32 ASFileGetEOF (ASFile aFile);
```

| | |
|------------------------|--|
| Description | Gets the current size of a file. |
| Parameters | <i>aFile</i> The file whose size is obtained. |
| Return Value | The size of the file. |
| Exceptions | fileErrlO |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileSetEOF |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileGetFileSys

```
ASFileSys ASFileGetFileSys (ASFile aFile);
```

| | |
|------------------------|--|
| Description | Gets the file system responsible for the specified open file. |
| Parameters | <p><i>aFile</i></p> <p>The open file whose file system is obtained.</p> |
| Return Value | The file's <i>ASFileSys</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | <u>ASFileGetFileSysByName</u> |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileGetMDFile

```
ASBool ASFileGetMDFile (ASFile fN, MDFile* pFileID,
    ASFileSys* pFileSys);
```

| | |
|------------------------|--|
| Description | Given an <i>ASFile</i> , returns the <i>fileSys</i> and the <i>MDFile</i> identification in that <i>fileSys</i> . This call is needed for a file system in a plug-in to be able to call the inner routines in another file system. |
| Parameters | <p><i>fN</i></p> <p>The <i>ASFile</i> for which the information is desired.</p> <p><i>pFileID</i></p> <p>(Filled by the method) Return pointer to the <i>MDFile</i> for this <i>fN</i>.</p> <p><i>pFileSys</i></p> <p>(Filled by the method) Return pointer to the file system for this <i>fN</i>.</p> |
| Return Value | <i>true</i> if <i>fN</i> is an open file, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileFromMDFile ASFileGetFileSys |
| Availability | Plug-ins: Available if PI_ACROSUPPORT_VERSION (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileGetPos

```
ASInt32 ASFileGetPos (ASFile aFile);
```

| | |
|------------------------|---|
| Description | Gets the current seek position in a file. This is the position at which the next read or write will begin. |
| Parameters | <i>aFile</i> The file in which to get the seek position. |
| Return Value | The current seek position. |
| Exceptions | fileErrIO |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileSetPos ASFileRead ASFileWrite |
| Availability | Plug-ins: Available if <i>PI_ACROSSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFilePushData

```
void ASFilePushData (ASFile aFile, const char* data,
    ASInt32 offset, ASInt32 length);
```

Description

Sends data from a file system implementation to an *ASFile*. The data may be for an multiread request call, or may be unsolicited.

This method can only be called from a file system. It must *not* be called by clients of *ASFile*, for example, by a caller that acquired an *ASFile* with [ASFileSysOpenFile](#).

Parameters

aFile

The file to which data is sent.

data

The data being pushed.

offset

File offset of the data.

length

The number of bytes of data.

Return Value

None

Exceptions

None

Notifications

None

Header File

ASCalls.h

Related Methods

[ASFileRead](#)

Availability

Plug-ins: Available if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileRead

```
ASInt32 ASFileRead (ASFile aFile, char* p, ASInt32 count);
```

| | |
|------------------------|---|
| Description | Reads data from a file, beginning at the current seek position. |
| Parameters | <p><i>aFile</i></p> <p>The file from which data is read.</p> <p><i>p</i></p> <p>(Filled by the method) A buffer into which data is read. The buffer must be able to hold at least <i>count</i> bytes.</p> <p><i>count</i></p> <p>The number of bytes to read.</p> |
| Return Value | The number of bytes actually read from the file. |
| Exceptions | fileErrIO fileErrUserRequestedStop fileErrBytesNotReady fileErrIOTimeout fileErrGeneral |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileWrite ASFileSetPos |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileReopen

```
ASInt32 ASFileReopen (ASFile aFile, ASUns16 mode);
```

| | |
|------------------------|--|
| Description | Attempts to reopen a file using the specified read/write mode. On some platforms, this may result in the file being closed and then reopened, and thus some error conditions may leave the file invalid. |
| Parameters | <p><i>aFile</i></p> <p>The file to reopen.</p> <p><i>mode</i></p> <p>An OR of the ASFile Open Modes.</p> |
| Return Value | Error code. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileSysOpenFile |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileSetEOF

```
void ASFileSetEOF (ASFile aFile, ASInt32 newFileSize);
```

| | |
|------------------------|---|
| Description | Changes the size of a file. The new size may be larger or smaller than the original size. |
| Parameters | <p><i>aFile</i></p> <p>The file whose size is changed.</p> <p><i>newFileSize</i></p> <p>The new size of <i>aFile</i>.</p> |
| Return Value | fileErrIO |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileGetEOF |
| Availability | Plug-ins: Available if <i>PI_ACROSSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileSetMode

```
ASUns32 ASFileSetMode (ASFile aFile, ASUns32 modeValue,
    ASUns32 modeMask);
```

| | |
|------------------------|---|
| Description | Gets and sets the mode flags for a file. |
| Parameters | <p><i>aFile</i></p> <p>The file for which to get or set the mode.</p> <p><i>modeValue</i></p> <p>The mode flag values to get or set, which are described in ASFileMode Flags.</p> <p><i>modeMask</i></p> <p>Mask for the mode flags to get or set.</p> |
| Return Value | The previous value of the mode. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileRead ASFileSysOpenFile |
| Example | <pre>/* Get the current mode */ ASUns32 currentMode; currentMode = ASFileSetMode(aFile, 0, 0); /* Set the mode */ ASFileSetMode(aFile, kASFileModeDoNotYieldIfBytesNotReady, kASFileModeDoNotYieldIfBytesNotReady); /* Clear the mode */ ASFileSetMode(aFile, 0, kASFileModeDoNotYieldIfBytesNotReady);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileSetPos

```
void ASFileSetPos (ASFile aFile, ASInt32 pos);
```

| | |
|------------------------|--|
| Description | Seeks to the specified position in a file. This is the position at which the next read or write will begin. |
| Parameters | <p><i>aFile</i></p> <p>The file in which to seek.</p> <p><i>pos</i></p> <p>The position to seek.</p> |
| Return Value | None |
| Exceptions | fileErrlO |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileGetPos ASFileRead ASFileWrite |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileWrite

```
ASInt32 ASFileWrite (ASFile aFile, const char* p,
    ASInt32 count);
```

| | |
|------------------------|---|
| Description | Writes data to a file, beginning at the current seek position. |
| Parameters | <p><i>aFile</i></p> <p>The file to which data is written.</p> <p><i>p</i></p> <p><i>(Filled by the method)</i> A buffer into which data is written. The buffer must be able to hold at least <i>count</i> bytes.</p> <p><i>count</i></p> <p>The number of bytes to write.</p> |
| Return Value | The number of bytes actually written to the file. |
| Exceptions | fileErrIO fileErrWrite |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileRead ASFileSetPos |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileSys

ASFileGetFileSysByName

```
ASFileSys ASFileGetFileSysByName (ASAtom name);
```

| | |
|------------------------|---|
| Description | Gets the file system that has the specified name. |
| Parameters | <p><i>name</i></p> <p>The <i>ASAtom</i> corresponding to the name of the file system to obtain. Use ASAtomFromString to convert a string to an <i>ASAtom</i>.</p> |
| Return Value | The specified file system. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileGetFileSys |
| Availability | Plug-ins: Available if <i>PL_ACROSSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileRegisterFileSys

```
ASBool ASFileRegisterFileSys (ASExtension extension,  
    ASFileSys fileSys);
```

| | |
|------------------------|---|
| Description | Allows an implementor to provide a file system for use by external clients. An implementation can register itself. An external client can locate the file system using ASFileGetFileSysByName . Note that an external client must know the format of the <i>ASPathName</i> for the <i>fileSys</i> . <i>fileSys</i> provides its name via the ASFileSysGetFileSysNameProc callback. If a file system with the same name is already registered, it returns <i>false</i> . |
| Parameters | <p><i>extension</i> The gExtensionID extension registering the <i>fileSys</i>.</p> <p><i>fileSys</i> The <i>fileSys</i> being registered.</p> |
| Return Value | <i>true</i> if <i>fileSys</i> is successfully registered, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileUnregisterFileSys ASFileGetFileSysByName |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileSysAcquireFileSysPath

```
ASPathName ASFileSysAcquireFileSysPath (ASFileSys oldFileSys,
ASPathName oldPathName, ASFileSys newFileSys);
```

Description

Converts an *ASPathName* from one file system to another. Returns an *ASPathName* on the new *ASFileSys* that refers to an image (possibly cached) of the file in the old *fileSys*. Use this call to get a local file that is an image of a remote file (in a URL file system, for example). This is needed by programs such as the Movie Player, because they can only work from local file-system files. The returned *fileName* may be a reference to a cache, so the file should be treated as read-only.

Because of the possibility of cache flushing, you must hold a copy of the remote file's *ASPathName* for the duration of use of the local file.

Do *not* remove the local file copy, since the *newFileSys* system does not know about the linkage to the remote (*oldFileSys*) file!

The source file does *not* have to be open.

This call is handled by *oldFileSys* if *oldFileSys* contains the appropriate procedure. Otherwise it is handled by copying the file. The source file is closed at the end of the copy if it was not open prior to the call.

Parameters

oldFileSys

The file system in which the *ASPathName* is currently valid.

oldPathName

The *ASPathname* in the current file system.

newFileSys

The file system to which the *ASPathName* is converted.

Return Value

The *ASPathName* in *newFileSys* or *NULL* if one can not be made.

After you are done using the *ASPathName*, you must release it with [ASFileSysReleasePath](#).

Exceptions

None

| | |
|------------------------|--|
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileSysCopyPath

```
ASPathName ASFileSysCopyPath (ASFileSys fileSys,  
ASPathName pathName);
```

| | |
|------------------------|--|
| Description | Copies the specified pathname (but does not copy the file specified by the pathname). The pathname must be in the correct form for the specified file system. This method may be used regardless of whether or not the file specified by the pathname is open. |
| Parameters | <i>fileSys</i> The file system containing <i>pathName</i> . It is a pointer to an ASFileSysRec structure. <i>pathName</i> The pathname to copy. |
| Return Value | Copy of <i>pathName</i> . After you are done using the ASPathName , you must release it using ASFileSysReleasePath . |
| Exceptions | genErrNoMemory |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileSysReleasePath |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileSysCreatePathName

```
ASPathName ASFileSysCreatePathName (const ASFileSys fileSys,
ASAtom pathSpecType, const void* pathSpec,
const void* mustBeZero);
```

| | |
|--------------------|--|
| Description | Creates an <i>ASPathName</i> based on the input type and <i>pathSpec</i> . Each <i>fileSys</i> implementation must publish the input types that it accepts. |
| Parameters | <p><i>fileSys</i></p> <p>The <i>ASFileSystem</i> you are trying to create an <i>ASPathName</i> for, or 0 for the default file system.</p> <p><i>pathSpecType</i></p> <p>An <i>ASAtom</i> from a string that specifies which type of <i>pathSpec</i> is being passed in.</p> <p>The Mac OS default <i>fileSys</i> accepts types of "Cstring", "FSSpec", or "SFReply". If it is a Cstring, it must be a full pathname separated by colons as in "VolumeName:FolderName1:FolderName2...:filename".</p> <p>The Windows default <i>fileSys</i> accepts types of Cstring as in "C:\directoryname\filename". Relative paths are also accepted as in "...sources\file.pdf".</p> <p>The Unix default <i>fileSys</i> accepts types of Cstring as in "/directoryname/filename". Relative paths are also accepted as in ".../sources/file.pdf".</p> <p><i>pathSpec</i></p> <p>The file specification from which to create an <i>ASPathName</i>. Relative paths are evaluated from the directory containing the executable (if used with the PDF Library), or the directory containing Acrobat (if used in a plug-in).</p> <p><i>mustBeZero</i></p> <p>Reserved for future use.</p> |

| | |
|------------------------|--|
| Return Value | <p>The newly-created pathname.</p> <p>In Mac OS, returns a non-<i>NULL</i> newly-created pathname when given a <i>NULL</i>-pointer to an FSSpec, or <i>NULL</i> when given a <i>NULL</i>-pointer to a Cstring or an SFReply.</p> <p>After you are done using the <i>ASPathName</i>, you must release it with ASFileSysReleasePath.</p> |
| Exceptions | genErrBadParm if the <i>pathSpecType</i> is not recognized. |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileSysCopyPath |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileSysDIPathFromPath

```
char* ASFileSysDIPathFromPath (ASFileSys fileSys,
    ASPathName path, ASPathName relativeToThisPath);
```

Description Converts a filename, specified as an *ASPathName*, to a device-independent pathname. This method calls the specified file system's [ASFileSysDiPathFromPathProc](#). After calling this method, you must release the path by calling [ASFileSysReleasePath](#).

Parameters

fileSys
The file system containing *pathname*. It is a pointer to an [ASFileSysRec](#) structure.

path
The path to convert.

relativeToThisPath
The pathname relative to which the device-independent pathname is specified. If *NULL*, the device-independent pathname will be an absolute, not a relative, pathname.

Return Value Device-independent pathname corresponding to the parameter values supplied. See Section 7.4 in the [Portable Document Format Reference Manual](#) for a description of the device-independent pathname format. This pathname may not be understood on another platform since drive specifiers may be prepended.

Exceptions None

Notifications None

Header File *ASCalls.h*

Related Methods [ASGetDefaultFileSys](#)
[ASFileSysPathFromDIPath](#)

Availability Plug-ins: Available if *PI_ACROSSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileSysOpenFile

```
ASInt32 ASFileSysOpenFile (ASFileSys fileSys,
    ASPathName pathName, ASUns16 mode, ASFile* fP);
```

| | |
|------------------------|---|
| Description | <p>Attempts to open a file in the specified file system, in the specified read/write/create mode. If the file is already open, the existing file handle is returned. The caller retains ownership of <i>pathName</i>.</p> <p>In Mac OS, when this method creates a file, the file's creator is set to "CARO" and its type is set to "PDF."</p> |
| Parameters | <p><i>fileSys</i></p> <p>The file system containing <i>pathname</i>. It is a pointer to an ASFileSysRec structure.</p> <p><i>pathname</i></p> <p>The file to open.</p> <p><i>mode</i></p> <p>An OR of the ASFile Open Modes.</p> <p><i>fP</i></p> <p>The <i>ASFile</i> that was opened. This value is filled by the method.</p> |
| Return Value | 0 if no errors occur while opening the file. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASPathFromPlatformPath ASFileClose ASGetDefaultFileSys |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileSysPathFromDIPath

```
ASPathName ASFileSysPathFromDIPath (ASFileSys fileSys,  
char* diPath, ASPathName relativeToThisPath);
```

Description

Converts a device-independent pathname to an *ASPathName*. This method can only be used for files that already exist (that is, it cannot be used to create a placeholder pathname for files that a plug-in intends to create in the future). This method calls the specified file system's [ASFileSysPathFromDIPathProc](#).

Parameters

fileSys

The file system on which the file resides. It is a pointer to an [ASFileSysRec](#) structure.

diPath

The device-independent pathname to convert. See Section 7.4 in the [Portable Document Format Reference Manual](#) for a description of the device-independent pathname format. This pathname may not be understood on another platform since drive specifiers may be prepended.

In Windows, you cannot specify a UNC pathname. You must have a file mounted on the file server. For example, the following path is valid:

```
/f/dirname/file.pdf
```

where *f* is \\server\people. The following does not work:

```
/server/people/dirname/file.pdf
```

relativeToThisPath

Pathname relative to which *diPath* is interpreted. If *NULL*, *diPath* is interpreted as an absolute pathname, not a relative pathname.

Return Value

ASPathName corresponding to the parameter values supplied. Returns *NULL* if *diPath* cannot be converted to an *ASPathName*, for example if the specified file does not already exist.

After you are done using the *ASPathName*, you must release it using [ASFileSysReleasePath](#).

Exceptions

[genErrNoMemory](#)

| | |
|------------------------|--|
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileSysDIPathFromPath ASFileSysReleasePath |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileSysReleasePath

```
void ASFileSysReleasePath (ASFileSys fileSys,
    ASPathName pathName);
```

| | |
|------------------------|--|
| Description | Decrements the internal reference count for <i>pathname</i> and disposes of the pathname (but not the file itself) if the reference count is zero. This does not result in any file-level operations, and is unaffected by whether or not there is an open file for this pathname. |
| Parameters | <p><i>fileSys</i></p> <p>The file system on which <i>pathName</i> resides. It is a pointer to an ASFileSysRec structure.</p> <p><i>pathName</i></p> <p>The pathname to release.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileSysRemoveFile

```
ASInt32 ASFileSysRemoveFile (ASFileSys fileSys,
    ASPathName pathName);
```

| | |
|------------------------|---|
| Description | Attempts to remove the directory link for <i>pathName</i> . <i>Warning:</i> If a file is already open for this <i>pathName</i> , the semantics of <i>ASFileSysRemoveFile</i> are file system-dependent. Make sure you've closed all <i>ASFiles</i> for <i>pathName</i> before calling <i>ASFileSysRemoveFile</i> . |
| Parameters | <p><i>fileSys</i></p> <p>The file system containing <i>pathname</i>. It is a pointer to an ASFileSysRec structure.</p> <p><i>pathname</i></p> <p>The file to delete.</p> |
| Return Value | 0 if no errors occur while removing the file. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileUnregisterFileSys

```
ASBool ASFileUnregisterFileSys (ASExtension extension,  
    ASFileSys fileSys);
```

| | |
|------------------------|--|
| Description | Allows a <i>fileSys</i> to be unregistered. In general, a <i>fileSys</i> is only unregistered by the extension that registered it. |
| Parameters | <i>extension</i> The gExtensionID extension unregistering the <i>fileSys</i> . <i>fileSys</i> The <i>fileSys</i> to unregister. |
| Return Value | <i>true</i> if <i>fileSys</i> successfully unregistered, <i>false</i> if any open files belong to the <i>fileSys</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileRegisterFileSys |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASGetDefaultFileSys

[ASFileSys](#) ASGetDefaultFileSys (void);

| | |
|------------------------|--|
| Description | Gets the default/standard file system implementation for a platform. |
| Parameters | None |
| Return Value | The platform's default file system. It is a pointer to an ASFileSysRec structure. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASPathFromPlatformPath |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASPathFromPlatformPath

[ASPathName](#) ASPathFromPlatformPath (void* platformPath);

| | |
|------------------------|--|
| Description | Converts a platform-specific pathname to an <i>ASPathName</i> . It can create an <i>ASPathName</i> from a file path where the file does <i>not</i> already exist. It works for Windows UNC pathnames as well. |
| Parameters | <p><i>platformPath</i></p> <p>Pointer to a platform-specific pathname. In Windows and Unix, it is a <i>NULL</i>-terminated string containing the full pathname with the appropriate path separators for each platform. In Mac OS, it is an <i>SFReply</i> structure.</p> |
| Return Value | <p>The <i>ASPathName</i> corresponding to <i>platformPath</i>.</p> <p>After you are done using the ASPathName, you must release it using ASFileSysReleasePath.</p> |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileSysReleasePath |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASStm

ASFileStmRdOpen

[ASStm](#) ASFileStmRdOpen ([ASFile](#) afile, ASUns16 bufSize);

| | |
|------------------------|---|
| Description | Creates a read-only <i>ASStm</i> from a file. The stream is seekable. |
| Parameters | <p><i>afile</i></p> <p>The open file to associate with the stream. The file must have been opened previously using ASFileSysOpenFile.</p> <p>Each open file has an unique <i>ASFile</i>. The <i>ASFile</i> value has meaning only to the common <i>ASFile</i> implementation and bears no relationship to platform-specific file objects.</p> <p><i>bufSize</i></p> <p>Length of data buffer, in bytes. If <i>bufSize</i> = 0, the default buffer size (currently 4kB) will be used. The default is generally reasonable. A larger buffer size should be used only when data in the file will be accessed in chunks larger than the default buffer.</p> <p>Although <i>bufSize</i> is passed as an <i>ASUns16</i>, it is treated internally as an <i>ASInt16</i>. As a result, buffer sizes above 32kB are not permitted.</p> |
| Return Value | The newly-created <i>ASStm</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileSysOpenFile ASStmRead ASStmClose CosNewStream ASMemStmRdOpen ASProcStmRdOpen |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFileStmWrOpen

[ASStm](#) ASFileStmWrOpen ([ASFile](#) aFile, ASUns16 bufSize);

| | |
|------------------------|--|
| Description | Creates a writable <i>ASStm</i> from a file. The stream is seekable. |
| Parameters | <p><i>aFile</i></p> <p>The open file to associate with the stream. The file must have been opened previously using ASFileSysOpenFile. Each open file has a unique <i>ASFile</i>. The <i>ASFile</i> value has meaning only to the common <i>ASFile</i> implementation and bears no relationship to platform-specific file objects.</p> <p><i>bufSize</i></p> <p>Length of a data buffer, in bytes. If <i>bufSize</i> = 0, the default buffer size (currently 4kB) is used. The default is generally reasonable. A larger buffer size should be used only when data in the file will be accessed in chunks larger than the default buffer. Although <i>bufSize</i> is passed as an ASUns16, it is treated internally as an ASInt16. As a result, buffer sizes above 32 kB are not permitted.</p> |
| Return Value | The newly-created <i>ASStm</i> . |
| Exceptions | genErrNoMemory |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASProcStmWrOpen ASFileStmRdOpen ASMemStmRdOpen ASProcStmRdOpen ASStmWrite ASStmRead ASStmClose ASFileSysOpenFile CosNewStream |

Example

```
ASStm OpenLogFile (char* pathname,
                  char* buf)
{
    ASPathName path = NULL;
    ASFile logFile = NULL;
    ASInt32 err = 0;
    ASUns16 mode = ASFILE_WRITE;
    ASStm writeLog;

    path = MakeCrossPlatformASPathName
        (pathname);
    err = ASFileSysOpenFile (ASGetDefaultFileSys
        (), path, mode, &logFile);
    if (err != 0) // returns 0 is no error
    {
        AAlertNote ("Unable to open data file for
            writing.");
        return false;
    }
    writeLog = ASFileStmWrOpen (logFile,
        sizeof(buf));
    return writeLog;
}
```

Availability

Plug-ins: Available if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|---------|
| Adobe PDF Library | — | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | — | M, W |
| Reader | — | M, W, U |

ASMemStmRdOpen

[ASStm](#) ASMemStmRdOpen (char* data, ASUns32 len);

| | |
|------------------------|---|
| Description | Creates a read-only <i>ASStm</i> from a memory-resident buffer. The stream is seekable. |
| Parameters | <p><i>data</i></p> <p>Buffer containing the data to read into the stream. This data buffer must not be disposed of until the <i>ASStm</i> is closed.</p> <p><i>len</i></p> <p>Length of data, in bytes.</p> |
| Return Value | The newly-created <i>ASStm</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASStmRead ASStmClose CosNewStream ASMemStmRdOpen ASProcStmRdOpen |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASProcStmRdOpen

[ASStm](#) ASProcStmRdOpen ([ASStmProc](#) readProc, void* clientData);

| | |
|------------------------|--|
| Description | Creates a read-only <i>ASStm</i> from an arbitrary data-producing procedure. The stream is not seekable. |
| Parameters | <p><i>readProc</i></p> <p>User-supplied callback that supplies the stream's data.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>readProc</i> each time it is called.</p> |
| Return Value | The newly-created <i>ASStm</i> . |
| Exceptions | genErrNoMemory |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASStmRead ASStmClose CosNewStream ASFileStmRdOpen ASMemStmRdOpen |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASProcStmWrOpen

```
ASStm ASProcStmWrOpen (ASStmProc writeProc,  
ASProcStmDestroyProc destroyProc, void* clientData);
```

| | |
|------------------------|--|
| Description | Creates an <i>ASStm</i> from an arbitrary data-producing procedure. The stream is not seekable. |
| Parameters | <p><i>writeProc</i></p> <p>User-supplied callback that provides the data for the stream.</p> <p><i>destroyProc</i></p> <p>User-supplied callback that destroys the specified <i>ASStm</i>. (Generally, this means deallocating the memory associated with the <i>ASStm</i>.)</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>writeProc</i> each time it is called.</p> |
| Return Value | The newly-created <i>ASStm</i> . |
| Exceptions | genErrNoMemory |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileStmWrOpen ASFileStmRdOpen ASMemStmRdOpen ASProcStmRdOpen ASStmWrite ASStmRead ASStmClose ASFileSysOpenFile CosNewStream |

Example

```
ACCB1 ASInt32 ACCB2 writeProc(char* data,
    ASInt32 nBytes, void* clientData);
{
    ...
}

ACCBPROTO1 void ACCBPROTO2 destroyProc
    (void* clientData);
{
    ...
}

ASStm writeLog;

writeProcPtr =
    ASCallbackCreateProto(ASStmProc,
    writeProc);
destroyProcPtr =
    ASCallbackCreateProto(ASProcStmDestroyProc,
    destroyProc);
writelog = ASProcStmWrOpen ( writeProc,
    destroyProc, void* clientData);
```

Availability

Plug-ins: Available if *PL_ACROSSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|---------|
| Adobe PDF Library | — | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | — | M, W |
| Reader | — | M, W, U |

ASStmClose

```
void ASStmClose (ASStm stm);
```

| | |
|------------------------|---|
| Description | Closes the specified stream. |
| Parameters | <i>stm</i> The stream to close. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFileStmRdOpen ASMemStmRdOpen ASProcStmRdOpen |
| Availability | Plug-ins: Available if <i>PL_ACROSSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASStmRead

```
ASInt32 ASStmRead (char* ptr, ASInt32 itemSize, ASInt32 nItems,
    ASStm stm);
```

| | |
|------------------------|---|
| Description | Reads data from <i>stm</i> into memory. |
| Parameters | <p><i>ptr</i></p> <p>Pointer to the start of the memory buffer into which data is read.</p> <p><i>itemSize</i></p> <p>Number of bytes in an item in the stream. See the description of <i>nItems</i> for further information.</p> <p><i>nItems</i></p> <p>Number of items to read. The amount of data read into the memory buffer will be <i>itemSize</i> × <i>nItems</i>, unless an <i>EOF</i> is encountered first.</p> <p>The relative values of <i>itemSize</i> and <i>nItems</i> really don't matter; the only thing that matters is their product. It is often convenient to set <i>itemSize</i> to 1, so that <i>nItems</i> is the number of bytes to read.</p> <p><i>stm</i></p> <p>The stream from which data is read.</p> |
| Return Value | The number of items (not bytes) read. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASStmWrite |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

ASStmWrite

```
ASInt32 ASStmWrite (const char* ptr, ASInt32 itemSize,
    ASInt32 nItems, ASStm stm);
```

Description

Writes data from a memory buffer into an *ASStm*.

You cannot use this method to change a PDF page contents stream—only a print stream.

Historically, this method was provided to allow plug-ins to write data into the print stream when printing to a PostScript printer (see the [PDDocWillPrintPage](#) notification). However, *ASStm* is a general purpose I/O mechanism in Acrobat, although only limited open and read/write methods are provided in the plug-in API. For instance, not all *ASStm* objects are seekable.

Parameters

ptr

Pointer to the start of the memory buffer from which data is read.

itemSize

Number of bytes in an item in the stream. See the description of *nItems* for additional information.

nItems

Number of items to write. The amount of data written into the stream will be $itemSize \times nItems$.

The relative values of *itemSize* and *nItems* really don't matter; the only thing that matters is their product. It is often convenient to set *itemSize* to 1, so that *nItems* is the number of bytes to read.

stm

The stream into which data is written.

Return Value

The number of items (not bytes) written.

Exceptions

None

Notifications

None

Header File

ASCalls.h

Related Methods

[ASStmRead](#)

Availability

Plug-ins: Available if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

Configuration

ASGetConfiguration

```
void* ASGetConfiguration (ASAtom key);
```

Description Gets information about the Acrobat viewer application under which the plug-in is running. Use this method if your plug-in's functionality depends on which Acrobat viewer it is running under.

Parameters *key*
Configuration parameter being requested. Must be one of the [ASGetConfiguration Selectors](#).
Use the *CanEdit* selector (not the *Product* selector) to determine the functionality available, because product names may change while functionality remains the same. Use the *Version* selector to determine the version of the Acrobat Developer Toolkit.

Return Value The return value's type depends on what was requested. See the [ASGetConfiguration Selectors](#). If an unknown value is passed as *key*, the value *UNDEFINED_CONFIGURATION_SELECTOR* is returned (see *CoreExpT.h*).

Exceptions None

Notifications None

Header File *CorCalls.h*

Related Methods None

Example

```
prodName = ASGetConfiguration(  
    ASAtomFromString("Product"))
```

Availability Plug-ins: Available if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

Errors

ASGetErrorString

```
const char* ASGetErrorString (ASInt32 errorCode, char* buffer,
    ASInt32 lenBuffer);
```

| | |
|------------------------|--|
| Description | Gets a string describing the specified error/exception. |
| Parameters | <p><i>errorCode</i></p> <p>The exception whose error string is obtained. This must be a full error code, built with the ErrBuildCode macro or a user-defined exception returned from ASRegisterErrorString. See Errors for a list of predefined exceptions.</p> <p><i>buffer</i></p> <p>(Filled by the method) Buffer into which the string is written.</p> <p><i>lenBuffer</i></p> <p>The number of characters that <i>buffer</i> can hold.</p> |
| Return Value | The error string, or <i>NULL</i> if the error is not known. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASGetExceptionErrorCode ASRegisterErrorString ASRaise |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASGetExceptionErrorCode

```
ASInt32 ASGetExceptionErrorCode (void);
```

| | |
|------------------------|---|
| Description | Gets the error code for the exception most recently raised. |
| Parameters | None |
| Return Value | Exception error code. This must be a full error code, built with the ErrBuildCode macro or a user-defined exception returned from ASRegisterErrorString . See Errors for a list of predefined exceptions. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CorCalls.h</i> |
| Related Methods | ASRaise ASGetErrorString ASRegisterErrorString |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASRaise

```
void ASRaise (ASInt32 error);
```

Description

Raises an exception. Plug-ins can raise any exception defined in the *AcroErr.h* header file using the [ErrBuildCode](#) macro, or can define their own exceptions using [ASRegisterErrorString](#). See [Errors](#) for a list of predefined exceptions.

If the code that calls *ASRaise* gets control as a result of a non-Acrobat event (such as a drag and drop event on some platforms), this method fails since there is no Acrobat viewer code in the stack to handle the exception.

Parameters

error

Error code for the exception to raise. Error codes have three parts: severity, system, and error number. Use [ErrBuildCode](#) to build an error code for an existing error.

Return Value

None

Exceptions

None

Notifications

None

Header File

CorCalls.h

Related Methods

[ASGetErrorString](#)
[ASRegisterErrorString](#)

Availability

Plug-ins: Available if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASRegisterErrorString

```
ASInt32 ASRegisterErrorString (ASErrSeverity severity,
    const char* errorString);
```

Description

Registers a new error and string. The error can be used to raise a plug-in-specific exception using [ASRaise](#). When the exception is raised, its error string can be retrieved using [ASGetErrorString](#) and reported to the user using [AAlertNote](#).

The error system is automatically forced to be *ErrSysXtn*. (See the list of [Error Systems](#).)

The error is automatically assigned an error code that is not used by any other plug-in (in the current implementation, the Acrobat viewer increments a counter each time any plug-in requests an error code, and returns the value of the counter). As a result, plug-ins cannot rely on being assigned the same error code each time the Acrobat viewer is launched.

Parameters

severity

The severity of the error being defined. Must be one of the [Severities](#).

errorString

The string describing the exception. This string is used by [ASGetErrorString](#). *errorString* is copied by *ASRegisterErrorString*; it may be freed by the plug-in after registering the error.

Return Value

The newly-created error code. Plug-ins should assign the error code returned by this method to a variable if they wish to use the error code later in the current session.

Exceptions

None

Notifications

None

Header File

ASCalls.h

Related Methods

[ASGetErrorString](#)
[ASRaise](#)

Availability

Plug-ins: Available if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

Fixed Point Math

ASCStringToFixed

[ASFixed](#) ASCStringToFixed (const char* s);

| | |
|------------------------|--|
| Description | Converts a Cstring to a fixed point number. Processes the string from left to right only until the first invalid character is located (for example, a-z, A-Z). |
| Parameters | s A Cstring to convert. |
| Return Value | Fixed number corresponding to s. Returns 0 if the string does not contain any valid number. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFixedToCString |
| Related Macros | FixedRoundToInt16 FixedRoundToInt32 FixedTruncToInt16 FixedTruncToInt32 FixedToFloat |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFixedDiv

```
ASFixed ASFixedDiv (ASFixed a, ASFixed b);
```

| | |
|------------------------|---|
| Description | Divides two fixed numbers. |
| Parameters | <i>a</i> , <i>b</i> The two numbers to divide. |
| Return Value | The quotient <i>a</i> / <i>b</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFixedMul |
| Related Macros | Fixed Numbers FixedRoundToInt16 FixedRoundToInt32 FixedTruncToInt16 FixedTruncToInt32 FixedToFloat FloatToFixed Int16ToFixed Int32ToFixed |
| Availability | Plug-ins: Available if <i>PI_ACROSSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFixedMatrixConcat

```
void ASFixedMatrixConcat (ASFixedMatrixP result,
    const ASFixedMatrixP m1, const ASFixedMatrixP m2);
```

| | |
|------------------------|---|
| Description | Multiplies two matrices. |
| Parameters | <p><i>result</i></p> <p>(Filled by the method) Pointer to matrix $m2 \times m1$. It is OK for <i>result</i> to point to the same location as either <i>m1</i> or <i>m2</i>.</p> <p><i>m1, m2</i></p> <p>Pointers to the ASFixedMatrix values for the two matrices to multiply.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFixedMatrixInvert ASFixedMatrixTransform ASFixedMatrixTransformRect |
| Related Macros | Fixed Numbers FixedRoundToInt16 FixedRoundToInt32 FixedTruncToInt16 FixedTruncToInt32 FixedToFloat FloatToFixed Int16ToFixed Int32ToFixed |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFixedMatrixInvert

```
void ASFixedMatrixInvert (ASFixedMatrixP result,
    const ASFixedMatrixP m);
```

| | |
|------------------------|--|
| Description | <p>Inverts a matrix.</p> <p>If a matrix is nearly singular (that is, has a determinant nearly zero), inverting and re-inverting the matrix may not yield the original matrix.</p> |
| Parameters | <p><i>result</i></p> <p>(Filled by the method) Pointer to <i>m</i>-1. It is OK for <i>result</i> to point to the same location as <i>m</i>.</p> <p><i>m</i></p> <p>Pointer to the ASFixedMatrix to invert.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFixedMatrixConcat ASFixedMatrixTransform ASFixedMatrixTransformRect |
| Related Macros | FixedRoundToInt16 FixedRoundToInt32 FixedTruncToInt16 FixedTruncToInt32 FixedToFloat |
| Availability | Plug-ins: Available if <i>PI_ACROSSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFixedMatrixTransform

```
void ASFixedMatrixTransform (ASFixedPointP result,
    const ASFixedMatrixP m, const ASFixedPointP p);
```

| | |
|------------------------|---|
| Description | Transforms the point <i>p</i> through the matrix <i>m</i> , puts result in <i>result</i> . <i>p</i> and <i>result</i> can point to the same place. |
| Parameters | <p><i>result</i></p> <p>(Filled by the method) Pointer to the ASFixedPoint containing the result of transforming <i>p</i> through <i>m</i>. It is OK for <i>result</i> to point to the same location as <i>m</i>.</p> <p><i>m</i></p> <p>Pointer to the ASFixedMatrix through which <i>p</i> is transformed.</p> <p><i>p</i></p> <p>Pointer to the ASFixedPoint representing the point to transform through <i>m</i>.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFixedMatrixTransformRect ASFixedMatrixConcat ASFixedMatrixInvert |
| Related Macros | Fixed Numbers FixedRoundToInt16 FixedRoundToInt32 FixedTruncToInt16 FixedTruncToInt32 FixedToFloat FloatToFixed Int16ToFixed Int32ToFixed |
| Availability | Plug-ins: Available if <i>PI_ACROSSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFixedMatrixTransformRect

```
void ASFixedMatrixTransformRect (ASFixedRectP result,
    const ASFixedMatrix* m, const ASFixedRect* rectIn);
```

| | |
|------------------------|---|
| Description | Transforms a rectangle through a matrix. |
| Parameters | <p><i>result</i></p> <p>(Filled by the method) Pointer to the ASFixedRect containing the smallest bounding box for the transformed rectangle. It is OK for <i>result</i> to point to the same location as <i>m</i>. <i>result</i> will always have <i>bottom</i> < <i>top</i> and <i>left</i> < <i>right</i>.</p> <p><i>m</i></p> <p>Pointer to the ASFixedMatrix containing the matrix through which <i>r</i> is transformed.</p> <p><i>rectIn</i></p> <p>Pointer to the ASFixedRect containing the rectangle to transform through <i>m</i>.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFixedMatrixTransform ASFixedMatrixConcat ASFixedMatrixInvert |
| Related Macros | Fixed Numbers FixedRoundToInt16 FixedRoundToInt32 FixedTruncToInt16 FixedTruncToInt32 FixedToFloat FloatToFixed Int16ToFixed Int32ToFixed |
| Availability | Plug-ins: Available if <i>PI_ACROSSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFixedMul

```
ASFixed ASFixedMul (ASFixed a, ASFixed b);
```

| | |
|------------------------|---|
| Description | Multiplies two fixed numbers. |
| Parameters | <i>a</i> , <i>b</i> The two numbers to multiply. |
| Return Value | The product of <i>a</i> and <i>b</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASFixedDiv |
| Related Macros | Fixed Numbers FixedRoundToInt16 FixedRoundToInt32 FixedTruncToInt16 FixedTruncToInt32 FixedToFloat FloatToFixed Int16ToFixed Int32ToFixed |
| Availability | Plug-ins: Available if <i>PI_ACROSSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASFixedToCString

```
void ASFixedToCString (ASFixed f, char* s, os_size_t maxLength,
    ASInt16 precision);
```

| | |
|------------------------|---|
| Description | Converts a fixed number to a CString. |
| Parameters | <p><i>f</i></p> <p>The fixed number to convert.</p> <p><i>s</i></p> <p><i>(Filled by the method)</i> The string corresponding to <i>f</i>.</p> <p><i>maxLength</i></p> <p>The maximum number of characters that <i>s</i> can contain.</p> <p><i>precision</i></p> <p>The number of digits to retain in the converted number.</p> <p><i>Note: The precision for Mac OS numbers is valid to 9 significant digits.</i></p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASCStringToFixed |
| Related Macros | Fixed Numbers FloatToFixed Int16ToFixed Int32ToFixed |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

HFTServer

ASExtensionMgrGetHFT

HFT ASExtensionMgrGetHFT (**ASAtom** name, ASUns32 version);

| | |
|------------------------|--|
| Description | Gets the Host Function Table (<i>HFT</i>) server that has the specified name and version. If you want to get one of the Acrobat viewer's built-in <i>HFTs</i> , use the predefined global variables for the HFTs instead of this method. |
| Parameters | <p><i>name</i></p> <p>The name of the <i>HFT</i> to obtain.</p> <p><i>version</i></p> <p>The version number of the <i>HFT</i> to obtain.</p> |
| Return Value | The specified <i>HFT</i> , or <i>NULL</i> if the <i>HFT</i> does not exist. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CorCalls.h</i> |
| Related Methods | HFTReplaceEntry |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

HFTDestroy

```
void HFTDestroy (HFT hft);
```

Description

Destroys an existing *HFT* by freeing all the *HFT*'s memory. Call this method only if you are *absolutely sure* that neither your plug-in nor any other plug-in will use the *HFT* again. Because this is usually impossible to know, plug-ins should not destroy *HFT*s. It is even dangerous to destroy an *HFT* at unload time, because the order in which plug-ins are unloaded is not specified.

Parameters

hft

The *HFT* to destroy.

Return Value

None

Exceptions

None

Notifications

None

Header File

ASCalls.h

Related Methods

[HFTNew](#)

Availability

Plug-ins: Available if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

HFTGetReplacedEntry

```
HFTEntry HFTGetReplacedEntry (HFT hft, Selector sel,
HFTEntry replacer);
```

Description

Gets the *HFTEntry* that was replaced by the specified *HFTEntry* in the specified entry. Plug-ins should generally not use this method directly, but use the [CALL_REPLACED_PROC](#) macro instead.

It is necessary to specify both a selector (the index of an entry in the *HFT*'s table of thunked function pointers) and an *HFTEntry* (a thunked function pointer) because a method may be replaced several times, and the various replacement methods are kept in a linked list. The selector determines which linked list is examined, and the *HFTEntry* determines the entry in the linked list to return.

Parameters

hft

The *HFT* in which a replaced entry is retrieved. See [HFTReplaceEntry](#) for more information.

sel

The selector whose previous value is obtained. See [HFTReplaceEntry](#) for more information.

replacer

The [HFTEntry](#) for which the previous value is obtained.

Return Value

The entry present prior to being replaced by *replacer*. Returns *NULL* if the entry has not been replaced.

Exceptions

None

Notifications

None

Header File

ASCalls.h

Related Methods

[ASExtensionMgrGetHFT](#)

Related Macros

[CALL_REPLACED_PROC](#)

Availability

Plug-ins: Available if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

HFTIsValid

```
ASBool HFTIsValid (HFT hft);
```

| | |
|------------------------|--|
| Description | Tests whether an <i>HFT</i> is valid. |
| Parameters | <i>hft</i> The <i>HFT</i> to test. |
| Return Value | <i>true</i> if <i>hft</i> is valid, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | None |
| Example | <code>result = HFTIsValid(gMyPluginHFT);</code> |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

HFTNew

HFT HFTNew (**HFTServer** hftServer, ASInt32 numSelectors);

| | |
|------------------------|--|
| Description | Creates a new <i>HFT</i> by calling the specified <i>HFT</i> server's HFTServerProvideHFTProc . |
| Parameters | <p><i>hftServer</i></p> <p>The <i>HFT</i> server for the <i>HFT</i> being created. The <i>HFT</i> server must have been created previously using HFTServerNew.</p> <p><i>numSelectors</i></p> <p>The number of entries in the new <i>HFT</i>. This determines the number of methods that the <i>HFT</i> can contain; each method occupies one entry.</p> |
| Return Value | The newly-created <i>HFT</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASExtensionMgrGetHFT HFTDestroy |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

HFTReplaceEntry

```
void HFTReplaceEntry (HFT hft, Selector sel, HFTEntry newEntry,  
ASUns32 flags);
```

Description

Replaces the specified entry in the specified *HFT*. This allows a plug-in to override and replace certain methods in the Acrobat viewer's API. See [Replaceable Methods](#) for a list of replaceable methods. This method can only be used within a plug-in's *importReplaceAndRegisterCallback* procedure.

Plug-ins can use the [REPLACE](#) macro instead of calling *HFTReplaceEntry* directly.

All plug-ins, and the Acrobat viewer itself, share a single copy of each *HFT*. As a result, when a plug-in replaces the implementation of a method, all other plug-ins and the Acrobat viewer also use the new implementation of that method. In addition, once a method's implementation has been replaced, there is no way to remove the new implementation without restarting the Acrobat viewer.

Parameters

hft

The *HFT* in which a method is replaced. Use [ASExtensionMgrGetHFT](#) to get the *HFT*, given its name. For the *HFTs* built into the Acrobat viewer, global variables containing the *HFTs* have been defined, so you can skip calling [ASExtensionMgrGetHFT](#) for these *HFTs*. See [HFTs](#) for a list of these global variables.

sel

The entry in the *HFT* to replace, derived from the method's name by appending *SEL*. For example, to replace [AVAlertNote](#), *sel* must be *AVAlertNoteSEL*.

newEntry

The function to replace the current one. The function pointer must be converted to an [HFTEntry](#) using the [ASCallbackCreateReplacement](#) macro.

flags

The new entry's properties. Currently, only [HFTEntryReplaceable](#) is defined.

| | |
|------------------------|--|
| Return Value | None |
| Exceptions | xmErrCannotReplaceSelector |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASExtensionMgrGetHFT HFTGetReplacedEntry |
| Related Macros | REPLACE ASCallbackCreateReplacement |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

HFTServerDestroy

```
void HFTServerDestroy (HFTServer hftServer);
```

| | |
|------------------------|--|
| Description | Destroys an <i>HFT</i> server. Call this method only if the <i>HFT</i> will not be used again. |
| Parameters | <i>hftServer</i> The <i>HFT</i> server to destroy. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | HFTServerNew |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

HFTServerNew

```
HFTServer HFTServerNew (const char* name,  
    HFTServerProvideHFTProc serverProc,  
    HFTServerDestroyProc destroyProc, void* clientData);
```

Description Creates a new Host Function Table (*HFT*) server. An *HFT* server is responsible for creating an instance of an *HFT* with the specified version number, and destroying the *HFT*.

Parameters

name
The new *HFT* server's name.

serverProc
(*Required*) User-supplied callback that provides an *HFT* when given a version number. This procedure is called by [HFTNew](#).

destroyProc
(*Optional*) User-supplied callback that destroys the specified *HFT* (generally, this means deallocating the memory associated with the *HFT*). This procedure is called by [HFTDestroy](#).

clientData
Pointer to user-supplied data to pass to the *HFT* server.

Return Value The newly-created *HFT* server.

Exceptions None

Notifications None

Header File *ASCalls.h*

Related Methods [HFTServerDestroy](#)
[HFTNew](#)

Availability Plug-ins: Available if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

Memory Allocation

ASfree

```
void ASfree (void* ptr);
```

| | |
|------------------------|--|
| Description | Frees the specified memory block. |
| Parameters | <i>ptr</i> The block of memory to free. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASmalloc ASrealloc |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASmalloc

```
void* ASmalloc (os_size_t nBytes);
```

| | |
|------------------------|--|
| Description | Allocates and returns a pointer to a memory block containing the specified number of bytes. |
| Parameters | <i>nBytes</i> The number of bytes for which space is allocated. |
| Return Value | Pointer to the allocated memory. Returns <i>NULL</i> on failure. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>ASCalls.h</i> |
| Related Methods | ASrealloc ASfree |
| Availability | Plug-ins: Available if <i>PI_ACROSUPPORT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

ASrealloc

```
void* ASrealloc (void* ptr, os_size_t newNBytes);
```

Description If possible, extends the given block and simply returns *ptr*. Otherwise, allocates a new block of *newNBytes* bytes, copies the contents at the old pointer into the new block, frees the old pointer, and returns the pointer to the new block. If a new block cannot be allocated, the call fails and *ptr* is not freed. Reallocating a block to a smaller size will never fail.

Parameters

ptr
The existing memory block.

newNBytes
The number of bytes the memory block must be able to hold.

Return Value Pointer to memory block.

Exceptions None

Notifications None

Header File *ASCalls.h*

Related Methods [ASmalloc](#)
[ASfree](#)

Availability Plug-ins: Available if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AV Layer

General

AVDestInfoDestroy

```
void AVDestInfoDestroy (AVDestInfo destInfo);
```

| | |
|------------------------|---|
| Description | Releases the memory associated with a destination. |
| Parameters | <p><i>destInfo</i></p> <p>The destination info object to destroy.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewToDestInfo |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020003 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVActionHandler

AVActionHandlerGetProcs

[AVActionHandlerProcs](#) AVActionHandlerGetProcs
([AVActionHandler](#) handler);

| | |
|------------------------|--|
| Description | Gets a structure containing pointers to the action handler's procedures. This method is useful when you want to subclass an already-registered action handler. |
| Parameters | <p><i>handler</i></p> <p>The action handler whose procedures are obtained.</p> |
| Return Value | Pointer to a structure that contains the action handler's callbacks. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppRegisterActionHandler |
| Example | <pre>/* Replace the DoProperties method of the Thread action handler */ AVActionHandlerProcs actHandler; #define Thread_K ASAtomFromString("Thread") actHandler = AVActionHandlerGetProcs(AVAppGetActionHandlerByType(Thread_K); actHandler->DoProperties = ASCallbackCreateProto(AVActionDoPropertiesProc, &myDoProperties); AVAppRegisterActionHandler(actHandler, NULL, (char*) ASAtomGetString(Thread_K), userName);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVActionHandlerGetType

[ASAtom](#) AVActionHandlerGetType ([AVActionHandler](#) handler);

Description Gets the *ASAtom* specifying what type of actions an action handler services. This is the same as the *pdfName* used when the action handler was registered using [AVAppRegisterActionHandler](#).

Parameters *handler*
The action handler whose type is obtained.

Return Value The *ASAtom* specifying what action types *handler* services.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppRegisterActionHandler](#)
[AVActionHandlerGetProcs](#)

Example

```
if (AVActionHandlerGetType(myActionHandler)
    == ASAtomFromString("GoToR")) {
    do something
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVActionHandlerGetUIName

```
const char* AVActionHandlerGetUIName (AVActionHandler handler);
```

Description Gets the string that was passed as the *userName* when the action handler was registered using [AVAppRegisterActionHandler](#).

Parameters *handler*
The action handler whose user interface name is obtained.

Return Value The user interface name of *handler*.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppRegisterActionHandler](#)

Example

```
if(strcmp(AVActionHandlerGetUIName(theHandler), "BalloonAction")){
    /* register BalloonAction */
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAlert

AVAlert

```
ASInt32 AVAlert (ASInt32 iconType, const char* msg,
    const char* button1, const char* button2,
    const char* button3, ASBool beep);
```

Description Displays an alert containing the specified message, icon, and one to three buttons with the specified titles. Sets the cursor to the arrow cursor.

You can replace this method with your own version, using [HFTReplaceEntry](#).

Parameters *iconType*
The icon to display. Must be one of the [AVAlert Icons](#).
Macintosh users: These constants are defined as per the standard Macintosh user interface guidelines.

msg
The message to display.

button1, button2, button3
Titles for up to three buttons. Rules:

- Use *NULL* to suppress a button's display.
- At least *button1* must be non-*NULL*.
- *button3* is not displayed if *button2* is *NULL*.

beep
true if it beeps, *false* otherwise.

Return Value The button number (1, 2, or 3) on which the user clicked.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [ASGetErrorString](#)
[AVAlertNote](#)
[AVAlertConfirm](#)
[AVDocAlert](#)

Example

```
ASInt32 choice = AVAlert(ALERT_CAUTION,
    "Reverting will discard all changes. Are
    you sure?", "OK", "Cancel", NULL,
    false);
```

Availability

Plug-ins: Available if *PL_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVAlertConfirm

```
ASBool AVAlertConfirm (const char* msg);
```

Description Displays a dialog box containing the specified message, an "OK," and "Cancel" buttons. Also beeps and sets the cursor to the arrow cursor.

Note: The implementation of AVAlertConfirm calls [AVAlert](#), which is a replaceable method. If [AVAlert](#) is replaced, AVAlertConfirm is also affected.

Parameters *msg*

The message to display.

Return Value Returns *true* if the user clicks "OK," *false* if the user clicks "Cancel."

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [ASGetErrorString](#)
[AVAlert](#)
[AVAlertNote](#)
[AVDocAlertConfirm](#)

Example

```
if(AVAlertConfirm("Are you sure you want to
delete all Bookmarks?")){

    PDBookmarkDestroy(PDDocGetBookmarkRoot(p
dDoc));
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAlertNote

```
void AVAlertNote (const char* msg);
```

Description Displays a dialog box containing an "OK" button and the specified message. Also beeps and sets the cursor to the arrow cursor.

Note: The implementation of AVAlertNote calls [AVAlert](#), which is a replaceable method. If [AVAlert](#) is replaced, AVAlertNote is also affected.

Parameters *msg*
The message to display.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [ASGetErrorString](#)
[AVAlert](#)
[AVAlertConfirm](#)
[AVDocAlertConfirm](#)

Example DURING

```
HANDLER
sprintf(buf, "Exception raised: %d, %s",
    ERRORCODE, ASGetErrorString(ERRORCODE));
AVAlertNote(buf);
END_HANDLER
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAnnotHandler

AVAnnotHandlerDeleteInfo

```
void AVAnnotHandlerDeleteInfo (AVAnnotHandler avanh,  
    AVAnnotHandlerInfo info);
```

| | |
|------------------------|---|
| Description | Delete the <i>AVAnnotHandlerInfo</i> associated with an annotation handler. |
| Parameters | <p><i>avanh</i></p> <p>The annotation handler.</p> <p><i>info</i></p> <p>The <i>AVAnnotHandlerInfo</i> associated with the annotation handler. The <i>info</i> contains the user interface name of annotation type and the platform-dependent bitmap used as the annotation icon.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAnnotHandlerGetInfo |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVAnnotHandlerGetInfo

[AVAnnotHandlerInfo](#) AVAnnotHandlerGetInfo ([AVAnnotHandler](#) avanh);

| | |
|------------------------|---|
| Description | Gets the information structure associated with an annotation handler. |
| Parameters | <p><i>avanh</i></p> <p>The annotation handler for which the information structure is needed.</p> |
| Return Value | The information structure associated with <i>avanh</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAnnotHandlerDeleteInfo |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVApp

AVAppBeginFullScreen

```
ASBool AVAppBeginFullScreen ( PDColorValue color );
```

Description Begins full-screen mode. In full-screen mode, all window borders, the menu bar, and the toolbar are hidden. All regions of the screen outside of the window boundary are painted with the specified color. *AVAppBeginFullScreen* is ignored if the application is already in full-screen mode.

Parameters *color*
The color to use for painting all regions of the screen outside of the window boundary.

Return Value *true* if the application enters full-screen mode, *false* if it is already in full-screen mode or the user selects "Cancel" from the dialog describing how to exit full-screen mode.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppEndFullScreen](#)
[AVAppDoingFullScreen](#)
[AVAppGetPreference](#)

Example

```
PDColorValue color;
if ( !AVAppDoingFullScreen() )
    AVAppBeginFullScreen( color );
else
    AVAppEndFullScreen();
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

AVAppBeginModal

```
void AVAppBeginModal (AVWindow window);
```

Description

Prepares the Acrobat viewer to display a modal window. For example, disables floating windows in Windows, where they are not automatically disabled. When you are done with the modal window, call [AVAppEndModal](#). Calling *AVAppBeginModal* does not make your window modal, it only informs the Acrobat viewer that you intend to display a modal window now.

Windows users: The parent of a modal window must be the application's window. If you display a second modal window from within a modal window, you must *not* call *AVAppBeginModal* a second time. Instead, make the first modal window the parent of the second.

Macintosh users: This method is a no-op.

UNIX users: This method is a no-op.

Parameters

window

The modal window to display.

Return Value

None

Exceptions

None

Notifications

None

Header File

AVCalls.h

Related Methods

[AVAppEndModal](#)
[AVAppModalWindowIsOpen](#)
[WinAppGetModalParent](#)

Example

```
AVWindow win = AVWindowNew();
if(!AVAppModalWindowIsOpen()){
    AVAppBeginModal(win); /* disable floating
    windows */
    AVAppShowWindow(win);
    AVAppEndModal(win);
}
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppCanQuit

```
ASBool AVAppCanQuit (void);
```

| | |
|------------------------|--|
| Description | The Acrobat viewer calls this method to obtain permission (programmatically, not using the user interface) when it wants to quit. To use this method, replace it with your own version, using HFTReplaceEntry . If you replace this method in UNIX, your replacement implementation must not have any user interface component (for example, dialog boxes). At the time your replacement is called, the Acrobat viewer owns the pointer and will not give it to your dialog, so the screen will freeze. |
| Parameters | None |
| Return Value | <i>true</i> if the Acrobat viewer can quit, <i>false</i> aborts the quit. The default version of this routine always returns <i>true</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | HFTReplaceEntry |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppDoingFullScreen

```
ASBool AVAppDoingFullScreen (void);
```

| | |
|------------------------|---|
| Description | Tests whether or not the application is running in full-screen mode. |
| Parameters | None |
| Return Value | <i>true</i> if application is currently in full-screen mode, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppBeginFullScreen AVAppEndFullScreen |
| Example | <pre>PDColorValue color; if (!AVAppDoingFullScreen()) AVAppBeginFullScreen(color); else AVAppEndFullScreen();</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppEndFullScreen

```
void AVAppEndFullScreen (void);
```

Description Ends full-screen mode. In full-screen mode, all window borders, the menu bar, and the toolbar are hidden. All regions of the screen outside of the window boundary are painted with the specified color.

Parameters None

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppBeginFullScreen](#)
[AVAppDoingFullScreen](#)
[AVAppGetPreference](#)

Example

```
PDColorValue color;
if (!AVAppDoingFullScreen())
    AVAppBeginFullScreen(color);
else
    AVAppEndFullScreen();
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppEndModal

```
void AVAppEndModal (void);
```

Description Informs the Acrobat viewer that a modal window is no longer being displayed.
Macintosh users: This method is a no-op.
UNIX users: This method is a no-op.

Parameters None

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppBeginModal](#)
[AVAppModalWindowIsOpen](#)
[WinAppGetModalParent](#)

Example

```
AVWindow win = AVWindowNew();
if (!AVAppModalWindowIsOpen()) {
    AVAppBeginModal(win); /* disable floating
    windows */
    AVAppShowWindow(win);
    AVAppEndModal(win);
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppEnumActionHandlers

```
void AVAppEnumActionHandlers ( AVActionEnumProc enumProc,
    void* clientData);
```

| | |
|------------------------|--|
| Description | Enumerates the currently installed action handlers. |
| Parameters | <p><i>enumProc</i></p> <p>User-supplied procedure to call once for each action handler.</p> <p>Enumeration ends if <i>enumProc</i> returns <i>false</i>.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>enumProc</i> each time it is called.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVActionHandlerGetProcs |
| Example | <pre>#define Thread_K ASAtomFromString("Thread") static ACCB1 ASBool ACCB2 myActionEnumerator(ASAtom type, char* userName, void* clientData) { if(type == Thread_K){ actHandler = AVActionHandlerGetProcs(AVAppGetActionHandlerByType(Thread_K)); origDoProperties = actHandler->DoProperties; return false; /* leave once we ve got what we need */ } return true; }</pre> <pre>AVAppEnumActionHandlers(ASCallbackCreateProto(AVActionEnumProc, &myActionEnumerator), NULL);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppEnumAnnotHandlers

```
void AVAppEnumAnnotHandlers (AVAnnotHandlerEnumProc enumProc,
    void* clientData);
```

| | |
|------------------------|---|
| Description | Enumerates all annotation handlers, calling a user-supplied procedure for each annotation handler. |
| Parameters | <p><i>enumProc</i></p> <p>User-supplied callback to call for each annotation handler.</p> <p>Enumeration ends if <i>enumProc</i> returns <i>false</i>.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>enumProc</i> each time it is called.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppGetAnnotHandlerByName AVAppRegisterAnnotHandler |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppEnumDocs

```
void AVAppEnumDocs (AVDocEnumProc enumProc, void* clientData);
```

Description Enumerates all open documents, calling a user-supplied procedure for each open document.

Parameters *enumProc*
User-supplied callback to call once for each open document. Note that *enumProc* must not close any documents.

Enumeration ends if *enumProc* returns *false*.

clientData

Pointer to user-supplied data to pass to *enumProc* each time it is called.

Return Value None

Exceptions Raises an exception if and only if *enumProc* raises an exception.

Notifications None

Header File *AVCalls.h*

Related Methods [PDEnumDocs](#)

Example

```
ACCB1 ASBool ACCB2 myAppEnumProc(AVDoc doc,
    void* clientData) {
    if (PDDocGetNumPages(AVDocGetPDDoc(doc))
        > 100)
        AAlertNote("Long document");
    return true;
}
...
AVAppEnumDocs(ASCallbackCreateProto
    (AVDocEnumProc, &myAppEnumProc), NULL);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppEnumTools

```
void AVAppEnumTools (AVToolEnumProc enumProc,
    void* clientData);
```

| | |
|------------------------|--|
| Description | Enumerates the registered tools, calling <i>enumProc</i> for each. |
| Parameters | <p><i>enumProc</i></p> <p>A user-supplied callback to call for each tool. Enumeration ends if <i>enumProc</i> returns <i>false</i>.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>enumProc</i> each time it is called.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppGetToolByName AVAppRegisterTool |
| Example | <pre>ACCB1 ASBool ACCB2 myToolEnumerator(AVTool tool, void* clientData) { return false; } AVAppEnumTools(ASCallbackCreateProto(AVToolEnumProc, &myToolEnumerator), NULL);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppEnumTransHandlers

```
void AVAppEnumTransHandlers (AVTransHandlerEnumProc enumProc,
    void* clientData);
```

| | |
|------------------------|---|
| Description | Enumerates the currently registered transition handlers. |
| Parameters | <p><i>enumProc</i></p> <p>User-supplied procedure to call once for each transition handler.</p> <p>Enumeration ends if <i>enumProc</i> returns <i>false</i>.</p> <p><i>clientData</i></p> <p>User-supplied data passed to <i>enumProc</i> each time it is called.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppGetTransHandlerByType AVAppRegisterTransHandler |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetActionHandlerByType

[AVActionHandler](#) AVAppGetActionHandlerByType ([ASAtom](#) type);

| | |
|------------------------|---|
| Description | Gets the action handler that services the specified action type. |
| Parameters | <p><i>type</i></p> <p>The action type whose handler is obtained.</p> |
| Return Value | The handler that services actions of the specified type. Returns <i>NULL</i> if no such handler is registered. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppRegisterActionHandler AVActionHandlerGetType AVActionHandlerGetType |
| Example | <pre>#define Thread_K ASAtomFromString("Thread") actHandler = AVActionHandlerGetProcs(AVAppGetActionHandlerByType(Thread_K));</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetActiveDoc

[AVDoc](#) AVAppGetActiveDoc (void);

| | |
|------------------------|--|
| Description | <p>Gets the frontmost document window. In UNIX, gets the window (containing an <i>AVDoc</i>) that got the last user event (Key or Button event). This method is often useful for menu or tool enable procs. The frontmost document may not be active, for example, the clipboard window may be active and over it.</p> <p>It is recommended that <i>AVAppGetActiveDoc</i> be called only from within menu item or tool button callbacks (AVExecuteProc, AVComputeEnabledProc, and AVComputeMarkedProc). For all other callbacks, the <i>AVDoc</i> should be derived from the parameters of the callback. For instance, obtain the <i>AVDoc</i> from an AVPageView parameter with AVPageViewGetAVDoc.</p> |
| Parameters | None |
| Return Value | <p>The frontmost document window, or <i>NULL</i> if no documents are open. <i>NULL</i> is also returned when a document is open but its invisible flag is set (see AVDocOpenParams) and may be returned while a document is being opened. This <i>AVDoc</i> may be in an external window, such as a Web browser's window.</p> |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppEnumDocs AVAppGetNumDocs |
| Example | <pre>/* Can be used as menu item enable proc */ static ACCB1 ASBool ACCB2 DocExistEnable(void* data){ return (AVAppGetActiveDoc() != NULL); }</pre> |
| Availability | <p>Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher.</p> |
| Available in: | |

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetActiveTool

[AVTool](#) AVAppGetActiveTool (void);

| | |
|------------------------|---|
| Description | Gets the active tool. |
| Parameters | None |
| Return Value | The active tool. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppSetActiveTool AVAppGetLastActiveTool AVAppGetDefaultTool |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetAnnotHandlerByName

[AVAnnotHandler](#) AVAppGetAnnotHandlerByName ([ASAtom](#) name);

| | |
|------------------------|--|
| Description | Gets the annotation handler that handles the specified annotation type. |
| Parameters | <i>name</i> Name of the requested annotation handler. The character string for the name can be converted to an <i>ASAtom</i> using ASAtomFromString . |
| Return Value | The annotation handler that services annotations of type <i>name</i> . Returns the default annotation handler if no handler services the specified annotation type. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppEnumAnnotHandlers |
| Example | <pre>#define Text_K ASAtomFromString("Text") AVAnnotHandler myAnnotHandler = AVAppGetAnnotHandlerByName(Text_K);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetCancelProc

[CancelProc](#) AVAppGetCancelProc (void** cancelProcClientDataP);

| | |
|------------------------|--|
| Description | <p>Gets the default <i>CancelProc</i>. A <i>CancelProc</i> is often passed methods that take a long time to run. At frequent intervals, the method calls the <i>CancelProc</i>. If the <i>CancelProc</i> returns <i>true</i>, the method cancels its operation.</p> <p>The default <i>CancelProc</i>'s behavior is platform-dependent.</p> <ul style="list-style-type: none"> • In Mac OS, an operation is canceled by a Command-period combination. • In Windows, an operation is canceled by the Escape key. |
| Parameters | <p><i>cancelProcClientDataP</i></p> <p>Data needed by <i>CancelProc</i>. This value is filled in by the method, and must be passed as the <i>clientData</i> whenever <i>CancelProc</i> is passed to a method.</p> |
| Return Value | The default <i>CancelProc</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | None |
| Example | <pre>CancelProc cm; void* cmData; cm = AVAppGetCancelProc(&cmData); PDDocInsertPages(AVDocGetPDDoc(doc), afterPage, pdDoc, 0, PDAllPages, NULL, pm, pmData, cm, cmData);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetDefaultTool

[AVTool](#) AVAppGetDefaultTool (void);

| | |
|------------------------|--|
| Description | Gets the default tool. Use this method, together with AVAppSetActiveTool , to restore the default tool any time you want. The default tool is the hand tool. |
| Parameters | None |
| Return Value | The default tool. Returns <i>NULL</i> if there is no default tool. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppSetActiveTool AVAppGetActiveTool AVAppGetLastActiveTool |
| Example | <pre>AVAppSetActiveTool(AVAppGetDefaultTool(), true);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetDocProgressMonitor

[ProgressMonitor](#) AVAppGetDocProgressMonitor
(void** progMonClientData);

| | |
|------------------------|---|
| Description | Gets the standard application progress monitor, which puts a thermometer in the message pane of the frontmost document window. This progress monitor can subsequently be passed to any API method requiring a progress monitor. |
| Parameters | <p><i>progMonClientData</i></p> <p><i>(Filled by the method)</i> Private data used by the progress monitor. This value is filled by <i>AVAppGetDocProgressMonitor</i>, and must be passed to any method that takes a progress monitor as a parameter. Because the value is filled by the method, it must point to a valid address to which data can be written.</p> |
| Return Value | The standard application progress monitor, or <i>NULL</i> if no documents are open. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | None |

Example

```
static ProgressMonitor progMon;
static void* monClientData;
progMon =
AVAppGetDocProgressMonitor(&monClientData);
if (progMon) {
    if (progMon->beginOperation)
        progMon->beginOperation(monClientData);
    if (progMon->setDuration)
        progMon->setDuration(len,
monClientData);
    if (progMon->setCurrValue)
        progMon->setCurrValue(0,
monClientData);
}
/* do long operation */
for(i=0;i<numBookmarks;i++){
    /* do the work */
    if (progMon->setCurrValue)
        progMon->setCurrValue(i,
monClientData);
}
if (progMon->beginOperation)
    progMon->beginOperation(monClientData);
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetLanguage

```
void AVAppGetLanguage (char* buffer);
```

| | |
|------------------------|---|
| Description | Gets the language in which the application's user interface is running. |
| Parameters | <p><i>buffer</i></p> <p>The user interface language. <i>buffer</i> must be able to contain four bytes. See the list of Language Codes for the allowed values.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppGetVersion |
| Example | <pre>char lang[4]; AVAppGetLanguage(lang); if(strcmp("ENU", lang)){ /* handle non-English language in special way */ }</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetLastActiveTool

[AVTool](#) AVAppGetLastActiveTool (void);

| | |
|------------------------|---|
| Description | Gets the tool that was active before the current tool became active. |
| Parameters | None |
| Return Value | The last active tool. If only one tool has ever been active, it is returned. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppGetActiveTool |
| Example | <pre>if (!toolPersistent) AVAppSetActiveTool(AVAppGetLastActiveTool(), true);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetMenubar

[AVMenubar](#) AVAppGetMenubar (void);

Description Gets the application's menu bar.

Parameters None

Return Value The menu bar.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuGetParentMenubar](#)
[AVMenubarGetMenuIndex](#)

Example `AVMenubar bar = AVAppGetMenubar();`

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetName

[ASAtom](#) AVAppGetName (void);

Description Gets the *ASAtom* corresponding to the application's name, that is, the name of the file containing the Acrobat viewer application. The user might have changed this, so don't use it to determine whether the application is Exchange or some other Acrobat viewer; use [ASGetConfiguration](#) instead.

Parameters None

Return Value The *ASAtom* returned can be converted to a string using [ASAtomGetString](#).

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [ASGetConfiguration](#)

Example

```
char buf[200];
const char*  appName;
ASAtom appNameAtom;

appNameAtom = AVAppGetName();
if (appNameAtom != ASAtomNull) {
    appName = ASAtomGetString(appNameAtom);
    sprintf(buf,"Application name is %s",
        appName);
    AVAlertNote(buf);
}
else /* This should >never< happen */
    AVAlertNote("Application name NULL!");
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

AVAppGetNumDocs

```
ASInt32 AVAppGetNumDocs (void);
```

| | |
|------------------------|---|
| Description | Gets the number of open document views. |
| Parameters | None |
| Return Value | The number of open <i>AVDocs</i> (zero if no documents are open). |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppGetActiveDoc |

Example

```
/* Might be used as a menu item s enable
   proc */
static ACCB1 ASBool ACCB2
DocExistEnable(void* data)
{
    if (AVAppGetNumDocs() <= 0)
        return false;
    else
        return true;
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetPreference

```
void* AVAppGetPreference (AVPrefsType preference);
```

| | |
|------------------------|---|
| Description | Gets the value of the specified built-in application preference. |
| Parameters | <p><i>preference</i></p> <p>The preference value to get. See the preference descriptions in AVPrefsType.</p> |
| Return Value | Depends on the preference requested. See the preference descriptions in AVPrefsType . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppSetPreference |
| Example | <pre>if (AVAppGetPreference(avpSkipWarnings) AAlertConfirm("Really?")){ AVAppSetPreference(avpShowToolBar, (void*)true); }</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetToolBar

```
AVToolBar AVAppGetToolBar (void);
```

| | |
|------------------------|--|
| Description | Gets Acrobat's main toolbar. A tool button may have its own toolbar called a <i>flyout</i> . Such a toolbar is <i>distinct</i> from the toolbar returned by <i>AVAppGetToolBar</i> . Create a flyout toolbar with AVToolBarNewFlyout , which you can attach to a tool button with AVToolButtonSetFlyout . Get a tool button's flyout (if it has one) with AVToolButtonGetFlyout . |
| Parameters | None |
| Return Value | The toolbar. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVToolBarEnumButtons AVToolBarGetFrame AVToolButtonGetFlyout AVToolButtonSetFlyout |
| Example | <pre>AVToolbar bar = AVAppGetToolBar();</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetToolByName

[AVTool](#) AVAppGetToolByName ([ASAtom](#) name);

| | |
|------------------------|--|
| Description | Gets the tool with the specified name. |
| Parameters | <p><i>name</i></p> <p>The <i>ASAtom</i> corresponding to the tool's name. The character string for the name can be converted to an <i>ASAtom</i> using ASAtomFromString. See Tool Names for a list of the names of the built-in tools.</p> |
| Return Value | The tool whose name was specified, or <i>NULL</i> if no such tool. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | ASAtomFromString |
| Example | <pre>AVTool noteTool = AVAppGetToolByName(ASAtomFromString("Note"));</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetTransHandlerByType

[AVTransHandler](#) AVAppGetTransHandlerByType ([ASAtom](#) name);

| | |
|------------------------|---|
| Description | Gets the transition handler registered for the specified transition type. |
| Parameters | <p><i>name</i></p> <p>Type of transition handler, which may be one of the types provided in the Acrobat viewer or a new type registered by a plug-in.</p> |
| Return Value | The transition handler for the type, or <i>NULL</i> if no transition handler is registered for that type. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppRegisterTransHandler |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppGetVersion

```
void AVAppGetVersion (ASInt16* majorP, ASInt16* minorP);
```

Description Gets the major and minor version numbers of the Acrobat viewer. For version 2.0, the major version number is 2 and the minor version number is 0. For version 2.1, the major version number is 2 and the minor version number is 1. For version 3.01, the major version number is 3 and the minor version number is 0.

Parameters

majorP
Pointer to the major version number.

minorP
Pointer to the minor version number.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppGetLanguage](#)

Example

```
ASInt16 major, minor;
AVAppGetVersion(&major, &minor);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppHandlePlatformEvent

```
ASBool AVAppHandlePlatformEvent (void* platformEvent);
```

| | |
|------------------------|---|
| Description | Handles a platform-specific event. |
| Parameters | <p><i>platformEvent</i></p> <p>A pointer to a platform-specific event structure.</p> |
| Return Value | <i>true</i> if the event was handled, <i>false</i> otherwise. |
| Exceptions | May raise exceptions, depending on the event. |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppHandleAppleEvent AVWindowHandlePlatformEvent |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVAppIsIdle

```
ASBool AVAppIsIdle (void);
```

Description

Returns *true* if the application is in a state in which it is safe to destroy a document that uses the multiread protocol. The multiread protocol is implemented on some platforms using a “yield” mechanism, which involves a transfer of information between two applications. This function returns *true* if there is no task executing in the application that could invoke such a transfer. Attempting to close a document during such a transfer can cause a deadlock event or a crash.

When a multiread protocol document is closed, the client must register an *AVAppIdle* task. During its idle proc, it must call [AVAppHandlePlatformEvent](#). If the call returns *true*, it is safe to call [AVDocClose](#). If it returns *false*, the client must retry at its next idle proc call. [AVAppHandlePlatformEvent](#) always returns *false* if invoked from an [AVExecuteProc](#).

This method does *not* protect against plug-ins that invoke the API outside of *AVAppIdle*, an [AVExecuteProc](#), or other explicit API methods. For example, if a Windows plug-in uses a Windows SetTimer event to call an API method, [AVAppHandlePlatformEvent](#) may erroneously return *true*. Therefore, plug-ins should always use the API methods.

Parameters

None

Return Value

true if it is safe to call [AVDocClose](#), *false* otherwise.

Exceptions

None

Notifications

None

Header File

AVCalls.h

Related Methods

[AVDocClose](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppModalWindowIsOpen

```
ASBool AVAppModalWindowIsOpen (void);
```

Description A plug-in should use this method to determine whether or not a modal window is open. There is a large (and ill-defined) group of actions that are illegal while a modal window is open, although these actions are not programmatically prevented by the Acrobat viewer. While a modal dialog is open, a plug-in must not open documents, change pages, change views, close documents, change tools, or do anything that might disrupt the user or Acrobat viewer.

Parameters None

Return Value *true* if a modal window is open, *false* otherwise.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppBeginModal](#)
[AVAppEndModal](#)

Example

```
if (!AVAppModalWindowIsOpen()) {
    AVAppBeginModal(win); /* disable
        floating windows */
    AVAppShowWindow(win);
    AVAppEndModal(win);
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppOpenHelpFile

```
ASBool AVAppOpenHelpFile (const char* fileName,
    ASBool doSearch);
```

Description Opens a specified help file. If the help file is not found, optionally opens a dialog box asking if the user wants to search for the help file.

Parameters *fileName*
Help file name. This is not a fully-qualified path name, but the name of an individual help file, such as "AcroHelp.pdf".

doSearch
If *true* and the help file is not found, displays a dialog box asking if the user wants to search for the help file. If *false*, returns *false* if the help file is not found.

Return Value *true* if the help file is found, *false* otherwise.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods None

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVAppRegisterActionHandler

```
void AVAppRegisterActionHandler
(AVActionHandlerProcs actionHandler, void* actionHandlerObj,
char* pdfName, char* userName);
```

| | |
|------------------------|---|
| Description | Adds a new action handler. |
| Parameters | <p><i>actionHandler</i></p> <p>A structure containing the callbacks for the action handler to register. This structure must <i>not</i> be freed after calling AVAppRegisterActionHandler.</p> <p><i>actionHandlerObj</i></p> <p>Pointer to user-supplied data to pass to the action handler's methods when they are invoked.</p> <p><i>pdfName</i></p> <p>The action type serviced by this handler, as it appears in the PDF file. Storage for this string may be released after the call. This string must not contain any whitespace characters (for example, spaces).</p> <p><i>userName</i></p> <p>The name of this action type as it should appear in the Acrobat viewer's user interface. Storage for this string may be released after the call.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVActionHandlerGetProcs AVAppEnumActionHandlers |

Example

```
AVActionHandlerProcs actHandler;
char userName[48];

actHandler = ASMalloc(
    sizeof(AVActionHandlerProcsRec));
if(!actHandler) return;
actHandler->size = sizeof(
    AVActionHandlerProcsRec);/* vital for
    future compatibility */
actHandler->Perform =
    ASCallbackCreateProto(
        AVActionPerformProc, &myPerform);

AVAppRegisterActionHandler(actHandler,
    NULL, "MultiAction",
    "Multiple Destination Link");
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppRegisterAnnotHandler

```
void AVAppRegisterAnnotHandler (AVAnnotHandler handler);
```

Description Registers a handler for an annotation subtype, replacing any previous handler that had been registered for that subtype. The annotation handler is not registered if its [AVAnnotHandlerGetTypeProc](#) returns *NULL*.

Parameters *handler*
Pointer to a structure containing the annotation handler's callbacks. This structure must *not* be freed after this call, but must be retained.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewInvertQuad](#)
[AVAppEnumAnnotHandlers](#)

Example

```
static AVAnnotHandlerRec myAH;

memset(&myAH, 0, sizeof(myAH));
myAH.size = sizeof(myAH); /* Vital for
    future compatibility */
myAH.flags = ANNOT_CLIP_TEXT_SELECTION;
myAH.GetType = ASCallbackCreateProto(
    AVAnnotHandlerGetTypeProc, &myGetType);
myAH.Draw = ASCallbackCreateProto(
    AVAnnotHandlerDrawProc, &myDraw);
myAH.GetAnnotViewBBBox =
    ASCallbackCreateProto(
        AVAnnotHandlerGetAnnotViewBBBoxProc,
        &myGetViewBBBox);
myAH.AdjustCursor = ASCallbackCreateProto(
    AVAnnotHandlerAdjustCursorProc,
    &myAdjustCursor);

AVAppRegisterAnnotHandler(&myAH);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppRegisterForPageViewAdjustCursor

```
void AVAppRegisterForPageViewAdjustCursor  
  (AVPageViewCursorProc cursorProc, void* data);
```

Description

Registers a user-supplied procedure to call each time the cursor is adjusted. If more than one procedure is registered, the procedures are called in the order that they were registered.

To unregister, you *must* use the same thunked value that was used to register; you cannot use a newly-thunked copy. To accomplish this, call [ASCallbackCreateProto](#) once before registering and use the value returned from this call both to register and unregister; do not call [ASCallbackCreateProto](#) a second time when unregistering.

Parameters

cursorProc

User-supplied callback to call each time the cursor is adjusted.

data

Pointer to user-supplied data to pass to *cursorProc* each time it is called.

Return Value

None

Exceptions

[genErrNoMemory](#)

Notifications

None

Header File

AVCalls.h

Related Methods

[AVAppUnregisterForPageViewAdjustCursor](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppRegisterForPageViewClicks

```
void AVAppRegisterForPageViewClicks  
  (AVPageViewClickProc clickProc, void* data);
```

| | |
|------------------------|---|
| Description | <p>Registers a user-supplied procedure to call each time a mouse click occurs. This is useful if a plug-in wishes to handle mouse clicks, and the plug-in is better implemented as something other than an annotation. If more than one routine is registered to receive mouse clicks, the most recently registered routine is called first.</p> <p>To unregister, you <i>must</i> use the same thunked value that was used to register; you cannot use a newly-thunked copy. To accomplish this, call ASCallbackCreateProto once before registering and use the value returned from this call both to register and unregister; do not call ASCallbackCreateProto a second time when unregistering.</p> |
| Parameters | <p><i>clickProc</i></p> <p>User-supplied callback to call each time a mouse click occurs.</p> <p>If the user-supplied callback returns <i>true</i>, it indicates that the procedure handled the mouse click, and the click should not be passed along to the Acrobat viewer or other plug-ins that registered to receive mouse clicks. If it returns <i>false</i>, the mouse click is passed to the Acrobat viewer or other plug-ins that registered to receive mouse clicks.</p> <p><i>data</i></p> <p>Pointer to user-supplied data to pass to <i>clickProc</i> each time it is called.</p> |
| Return Value | None |
| Exceptions | genErrNoMemory |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppUnregisterForPageViewClicks |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppRegisterForPageViewDrawing

```
void AVAppRegisterForPageViewDrawing  
(AVPageViewDrawProc proc, void* data);
```

Description

Registers a user-supplied procedure to call each time a window is drawn, after the page's contents and annotations have been drawn. This allows plug-ins to draw whatever they wish to on top of pages. If more than one procedure is registered, they are called in a last-in, last-out order. *proc* is called each time the page view changes (scrolls, zooms, goes to a different page).

To unregister, you *must* use the same thunked value that was used to register; you cannot use a newly-thunked copy. To accomplish this, call [ASCallbackCreateProto](#) once before registering and use the value returned from this call both to register and unregister; do not call [ASCallbackCreateProto](#) a second time when unregistering.

Parameters

proc

User-supplied callback to call each time a window is drawn.

data

Pointer to user-supplied data to pass to *proc* each time it is called.

Return Value

None

Exceptions

[genErrNoMemory](#)

Notifications

None

Header File

AVCalls.h

Related Methods

[AVAppUnregisterForPageViewDrawing](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppRegisterIdleProc

```
void AVAppRegisterIdleProc (AIdleProc idleProc,
    void* clientData, ASUns32 period);
```

Description

Registers a user-supplied procedure to call “regularly” when the Acrobat viewer is otherwise idle. If more than one idle procedure is registered, they are all called in a round robin order. The registered idle procs may be called when the Acrobat viewer is not the frontmost application. In addition, in Mac OS, the registered idle procs receive idle events any time a movable modal dialog or modal *AVWindow* is displayed, but not a system-modal one. Use [AVAppModalWindowIsOpen](#) if you wish to determine whether or not a modal window is open.

To unregister, you *must* use the same thunked value that was used to register; you cannot use a newly-thunked copy. To accomplish this, call [ASCallbackCreateProto](#) once before registering and use the value returned from this call both to register and unregister; do not call [ASCallbackCreateProto](#) a second time when unregistering.

Parameters

idleProc

User-supplied callback to call at idle time.

clientData

Pointer to user-supplied data to pass to *idleProc* each time it is called.

period

Minimum time between calls to *idleProc*. *idleProc* will not be called any more frequently than *period*, but it may be called less frequently. *period* is specified in ticks (one tick is 1/60 of a second).

Return Value

None

Exceptions

None

Notifications

None

Header File

AVCalls.h

Related Methods

[AVAppUnregisterIdleProc](#)
[AVAppModalWindowIsOpen](#)

Example

```
ACCB1 void ACCB2 myIdleProc(void* data)
{
    /* do some stuff */
}
AVAppRegisterIdleProc(
    ASCallbackCreateProto(AVIdleProc,
        &myIdleProc), NULL, 15);
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppRegisterNotification

```
void AVAppRegisterNotification (NSElector nsel,
    ASExtension owner, void* proc, void* clientData);
```

Description

Registers a user-supplied procedure to call when the specified event occurs.

Many notifications appear in *Will/Did* pairs, for example [AVDocWillPerformAction](#) and [AVDocDidPerformAction](#). It is possible that an operation may fail after the *Will* notification and before the *Did* notification. When this occurs, the *Did* notification is still broadcast, but the *err* parameter in the *Did* notification is nonzero, and represents the error that occurred. When *err* is nonzero, the other parameters are not necessarily valid. Always check *err* in a *Did* notification before using the other parameters.

When calling [AVAppUnregisterNotification](#) to unregister for a notification, you *must* pass the *proc*, *clientData*, and *owner* that were used when the notification was registered using *AVAppRegisterNotification*. You *must* use the same thunked value that was used to register; you cannot use a newly-thunked copy. To accomplish this, call [ASCallbackCreateNotification](#) once before registering and use the value returned from this call both to register and unregister; do not call [ASCallbackCreateNotification](#) a second time when unregistering.

Parameters

nsel

Notification type. Must be one of the notification selectors (see [Notifications](#)). The notification selector is the name of the notification with the characters *NSEL* appended. For example, the selector for [AVDocDidOpen](#) is *AVDocDidOpenNSEL*.

owner

The [gExtensionID](#) extension registering the notification.

proc

User-supplied callback to call when the notification occurs. Its declaration depends on the notification type (see [Notifications](#)). Remember to use [ASCallbackCreateNotification](#) to convert *proc* to a callback before passing it to *AVAppRegisterNotification*.

clientData

Pointer to user-supplied data to pass to *proc* each time it is called.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppUnregisterNotification](#)

Example

```
ASCallback myCallback;

myCallback = ASCallbackCreateNotification(
    AVDocDidOpen, &CalcAVDocDidOpen);
AVAppRegisterNotification(AVDocDidOpenNSEL,
    gExtensionID, myCallback, NULL);
...
/* Unregister for the notification */
AVAppUnregisterNotification(
    AVDocDidOpenNSEL, gExtensionID,
    myCallback, NULL);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppRegisterTool

```
void AVAppRegisterTool (AVTool tool);
```

| | |
|------------------------|--|
| Description | Registers the specified tool with the Acrobat viewer. The tool is not registered if its GetTypeProcType callback returns <i>NULL</i> . |
| Parameters | <p><i>tool</i></p> <p>Structure containing the tool's callbacks. This structure must <i>not</i> be freed after calling <i>AVAppRegisterTool</i>, but must be retained.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppEnumTools AVAppSetActiveTool |
| Example | <pre>static AVToolRec myTool; { memset(&myTool, 0, sizeof(AVToolRec)); myTool.size = sizeof(AVToolRec); myTool.Activate = ASCallbackCreateProto(ActivateProcType, &myToolActivate); myTool.GetType = ASCallbackCreateProto(GetTypeProcType, &myToolGetType); myTool.AdjustCursor = ASCallbackCreateProto(AdjustCursorProcType, &myToolAdjustCursor); myTool.DoClick = ASCallbackCreateProto(DoClickProcType, &myToolDoClick); myTool.ComputeEnabled = myToolIsEnabledCallback; /* already callbackCreated */ myTool.computeEnabledData = NULL; AVAppRegisterTool(&myTool); }</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppRegisterTransHandler

```
void AVAppRegisterTransHandler (AVTransHandler avth);
```

| | |
|------------------------|---|
| Description | Adds a new transition handler. |
| Parameters | <p><i>avth</i></p> <p>An AVTransHandler structure containing pointers to the transition handler's functions. This structure must <i>not</i> be freed after calling AVAppRegisterTransHandler.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppEnumTransHandlers AVAppGetTransHandlerByType |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppSetActiveTool

```
void AVAppSetActiveTool (AVTool tool, ASBool persistent);
```

| | |
|------------------------|--|
| Description | Sets the active tool. Does nothing if the specified tool is not currently enabled. Whether a tool is enabled or not is determined by the AVComputeEnabledProc callback in the AVTool structure. If this callback is <i>NULL</i> , the tool is always enabled. |
| Parameters | <p><i>tool</i></p> <p>The tool to set as the active tool.</p> <p><i>persistent</i></p> <p>A flag that indicates a preference as to whether or not the tool stays active after it is used. <i>true</i> is a hint that the tool should, if possible, stay active for an arbitrary number of "operations" (whatever that happens to be) rather than doing a "one shot" operation and restoring the prior active tool. Persistence is not enforced by the Acrobat viewer. It is up to a one-shot tool to restore the previously active tool (determined using AVAppGetLastActiveTool), or to restore the default tool (determined using AVAppGetDefaultTool) if there was no previously active tool.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppGetActiveTool AVAppGetLastActiveTool AVAppGetDefaultTool |
| Example | <pre>if (!toolPersistent) AVAppSetActiveTool(AVAppGetLastActiveTool(), true);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppSetPreference

```
void AVAppSetPreference (AVPrefsType preference,
    void* newValue);
```

| | |
|------------------------|---|
| Description | Sets the value of the specified built-in application preference. The preference values are automatically saved to disk when a new value is set. |
| Parameters | <p><i>preference</i></p> <p>The preference value to set. See the preference descriptions in AVPrefsType.</p> <p><i>newValue</i></p> <p>The new value for a preference. The value must be cast to the appropriate data type (see AVPrefsType).</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppGetPreference AVDocSetSplitterPosition AVMenuItemNew |
| Example | <pre>if (AVAppGetPreference(avpSkipWarnings) AAlertConfirm("Really?")) { AVAppSetPreference(avpShowToolBar, (void *)true); }</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppUnregisterForPageViewAdjustCursor

```
void AVAppUnregisterForPageViewAdjustCursor  
    (AVPageViewCursorProc cursorProc);
```

Description Unregisters a user-supplied procedure that was called each time the cursor is adjusted.

To unregister, you *must* use the same thunked value that was used to register; you cannot use a newly-thunked copy. To accomplish this, call [ASCallbackCreateProto](#) once before registering and use the value returned from this call both to register and unregister; do not call [ASCallbackCreateProto](#) a second time when unregistering.

Parameters *cursorProc*

User-supplied callback to call each time the cursor is adjusted.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppRegisterForPageViewAdjustCursor](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppUnregisterForPageViewClicks

```
void AVAppUnregisterForPageViewClicks  
    (AVPageViewClickProc clickProc);
```

Description Unregisters a user-supplied procedure that was called each time a mouse click occurs.

To unregister, you *must* use the same thunked value that was used to register; you cannot use a newly-thunked copy. To accomplish this, call [ASCallbackCreateProto](#) once before registering and use the value returned from this call both to register and unregister; do not call [ASCallbackCreateProto](#) a second time when unregistering.

Parameters *clickProc*

User-supplied callback to call each time a mouse click occurs.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppRegisterForPageViewClicks](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppUnregisterForPageViewDrawing

```
void AVAppUnregisterForPageViewDrawing  
  ( AVPageViewDrawProc proc );
```

Description Unregisters a user-supplied procedure that was called each time a window is drawn, after the page's contents and annotations have been drawn.

To unregister, you *must* use the same thunked value that was used to register; you cannot use a newly-thunked copy. To accomplish this, call [ASCallbackCreateProto](#) once before registering and use the value returned from this call both to register and unregister; do not call [ASCallbackCreateProto](#) a second time when unregistering.

Parameters *proc*
User-supplied callback to call each time a window is drawn.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppRegisterForPageViewDrawing](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppUnregisterIdleProc

```
void AVAppUnregisterIdleProc (AIdleProc idleProc,
    void* clientData);
```

Description Unregisters a user-supplied procedure that was called “regularly” when the Acrobat viewer is otherwise idle. To unregister, you *must* use the same thunked value that was used to register; you cannot use a newly-thunked copy. To accomplish this, call [ASCallbackCreateProto](#) once before registering and use the value returned from this call both to register and unregister; do not call [ASCallbackCreateProto](#) a second time when unregistering.

Parameters

idleProc
User-supplied callback to call at idle time.

clientData
Pointer to user-supplied data to pass to *idleProc* each time it is called.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppRegisterIdleProc](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVAppUnregisterNotification

```
void AVAppUnregisterNotification (NSElector nsel,
    ASExtension owner, void* proc, void* clientData);
```

Description

Unregisters a user-supplied procedure that was called when the specified event occurs.

When calling *AVAppUnregisterNotification*, you must pass the *proc*, *clientData*, and *owner* that were used when the notification was registered using [AVAppRegisterNotification](#). You *must* use the same thunked value that was used to register; you cannot use a newly-thunked copy. To accomplish this, call [ASCallbackCreateNotification](#) once before registering and use the value returned from this call both to register and unregister; do not call [ASCallbackCreateNotification](#) a second time when unregistering.

Parameters

nsel

Notification type. Must be one of the notification selectors (see [Notifications](#)). The notification selector is the name of the notification with the characters *NSEL* appended. For example, the selector for [AVDocDidOpen](#) is *AVDocDidOpenNSEL*.

owner

The [gExtensionID](#) extension registering the notification.

proc

User-supplied callback to call when the notification occurs.

You must use the *same* callback that you called [AVAppRegisterNotification](#) with.

clientData

Pointer to user-supplied data to pass to *proc* each time it is called.

Return Value

None

Exceptions

None

Notifications

None

Header File

AVCalls.h

Related Methods [AVAppRegisterNotification](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVHasAuxDataHandler

```
ASBool AVHasAuxDataHandler (ASAtom auxDataType);
```

| | |
|------------------------|--|
| Description | Indicates whether a handler exists for the data type. |
| Parameters | <i>auxDataType</i> Name of data handler being queried. |
| Return Value | <i>true</i> if a handler is registered, <i>false</i> if a handler is not registered. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocSendAuxData AVRegisterAuxDataHandler AVUnregisterAuxDataHandler |
| Example | AVRegisterAuxDataHandler |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVRegisterAuxDataHandler

```
ASBool AVRegisterAuxDataHandler (ASExtension extension,
    ASAtom auxDataType, AVAuxDataHandler handler);
```

| | |
|------------------------|--|
| Description | Registers an <i>AuxDataHandler</i> . A handler can be registered for more than one kind of auxiliary data. The type of data is passed to the <i>AuxDataHandler</i> at <i>Perform</i> time so that it can distinguish what type of data it is receiving. |
| Parameters | <p><i>extension</i></p> <p>The gExtensionID extension registering the handler.</p> <p><i>auxDataType</i></p> <p>The type of data that the handler accepts.</p> <p><i>handler</i></p> <p>The handler to register.</p> |
| Return Value | <i>true</i> if the handler was registered, <i>false</i> if the handler could not be unregistered. For example, if another handler is already registered for this time. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVHasAuxDataHandler AVUnregisterAuxDataHandler |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVUnregisterAuxDataHandler

```
ASBool AVUnregisterAuxDataHandler (ASExtension extension,  
    ASAtom auxDataType, AVAuxDataHandler handler);
```

| | |
|------------------------|--|
| Description | Unregisters a previously registered <i>AuxDataHandler</i> . |
| Parameters | <p><i>extension</i></p> <p>The gExtensionID extension unregistering the handler.</p> <p><i>auxDataType</i></p> <p>Type of data for the handler being unregistered.</p> <p><i>handler</i></p> <p>The handler to unregister.</p> |
| Return Value | <i>true</i> if the handler was unregistered, <i>false</i> if the handler could not be unregistered. For example, returns <i>false</i> if the handler was never registered. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVHasAuxDataHandler AVRegisterAuxDataHandler |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVCrypt

AVAuthOpen

```
ASBool AVAuthOpen (PDDoc pdDoc);
```

| | |
|------------------------|---|
| Description | Determines if a user is authorized to open a document. |
| Parameters | <p><i>pdDoc</i></p> <p>The document whose password is obtained from the user.</p> |
| Return Value | <i>true</i> if the user is authorized to open the document, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | PDDocAuthorize |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVCryptDoStdSecurity

```
ASBool AVCryptDoStdSecurity (PDDoc pdDoc, void* secData);
```

Description Displays a security dialog to the user, allowing the user to change a document's print, edit, and copy permissions.

This method is present in the API so that other security handlers can use it, if they choose, as a standard way to display/obtain permissions from a user.

Parameters *pdDoc*

The document for which new security data is obtained.

secData

Pointer to data representing the permissions the user selected in the dialog.

Return Value *true* if the user OK'ed the dialog, *false* if the user canceled the dialog.

Exceptions None

Notifications None

Header File *PICrypt.h*

Related Methods [PDDocAuthorize](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

AVCryptGetPassword

```
ASBool AVCryptGetPassword (PDDoc pdDoc, PDPerms permWanted,
    void** authDataP);
```

Description Displays the standard dialog box that lets a user enter a password. This method does not check the password; that function is performed by [PDDocAuthorize](#).

This method is present in the API so that other security handlers can use it, if they choose, as a standard way to obtain a password from a user.

Parameters *pdDoc*
The document whose password is obtained from the user.

permWanted
The permissions requested. Must be an OR of the [PDPerms](#) values.

authDataP
Pointer to an authorization data structure into which the password is placed.

Return Value *false* if the user canceled the dialog, *true* otherwise.

Exceptions None

Notifications None

Header File *PICrypt.h*

Related Methods [PDDocAuthorize](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDoc

AVDocAlert

```
ASInt32 AVDocAlert (AVDoc doc, ASInt32 iconType,
    const char* msg, const char* button1, const char* button2,
    const char* button3, ASBool beep);
```

Description

Displays an alert containing the specified message, icon, and one to three buttons with the specified titles. Sets the cursor to the arrow cursor.

This method behaves similarly to [AAlert](#). On the Windows platform, the modal parent for *doc* (as returned by [WinAppGetModalParent](#)) is used as the owner window of the message dialog box.

Parameters

doc

(*Windows only*) The *AVDoc* whose modal parent is used as the owner window of the message dialog box.

iconType

The icon to display. Must be one of the [AAlert Icons](#).

Macintosh users: These constants are defined as per the standard Macintosh user interface guidelines.

msg

The message to display.

button1, button2, button3

Titles for up to three buttons. Rules:

- Use *NULL* to suppress a button's display.
- At least *button1* must be non-*NULL*.
- *button3* is not displayed if *button2* is *NULL*.

beep

true if it beeps, *false* otherwise.

Return Value

The button number (1, 2, or 3) on which the user clicked.

Exceptions

None

Notifications

None

Header File

AVCalls.h

Related Methods [AVAlert](#)
[AVDocAlertConfirm](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVDocAlertConfirm

```
ASBool AVDocAlertConfirm (AVDoc doc, const char* msg);
```

Description Displays a dialog box containing the specified message, an “OK,” and “Cancel” buttons. Also beeps and sets the cursor to the arrow cursor.

This method behaves similarly to [AVAlertConfirm](#). On the Windows platform, the modal parent for *doc* (as returned by [WinAppGetModalParent](#)) is used as the owner window of the message dialog box.

Parameters *doc*
(*Windows only*) The *AVDoc* whose modal parent is used as the owner window of the message dialog box.

msg
The message to display.

Return Value *true* if the user clicked “OK,” *false* if the user clicked “Cancel.”

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAlertConfirm](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVDocAlertNote

```
void AVDocAlertNote (AVDoc doc, const char* msg);
```

Description

Displays a dialog box containing an “OK” button and the specified message. Also beeps and sets the cursor to the arrow cursor.

This method behaves similarly to [AAlertNote](#). On the Windows platform, the modal parent for *doc* (as returned by [WinAppGetModalParent](#)) is used as the owner window of the message dialog box.

Parameters

doc

(*Windows only*) The *AVDoc* whose modal parent is used as the owner window of the message dialog box.

msg

The message to display.

Return Value

None

Exceptions

None

Notifications

None

Header File

AVCalls.h

Related Methods

[AAlertNote](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVDocAlertYesNo

```
ASInt32 AVDocAlertYesNo (AVDoc doc, ASInt32 iconType,
    const char* msg, ASBool cancel, ASBool beep);
```

Description

Similarly to [AVDocAlert](#), displays an alert dialog box containing the specified message, icon, and two or three buttons with the titles "Yes", "No", and (optionally) "Cancel". Sets the cursor to the arrow cursor. This method

This method behaves similarly to [AVAlert](#). On the Windows platform, the modal parent for *doc* (as returned by [WinAppGetModalParent](#)) is used as the owner window of the message dialog box.

Parameters

doc

(*Windows only*) The *AVDoc* whose modal parent is used as the owner window of the message dialog box.

iconType

The icon to display. Must be one of the [AVAlert Icons](#).

Macintosh users: These constants are defined as per the standard Macintosh user interface guidelines.

msg

The message to display.

cancel

true if cancel button should be provided, *false* otherwise.

beep

true if it beeps, *false* otherwise.

Return Value

The button number (1, 2, or 3) on which the user clicked.

Exceptions

None

Notifications

None

Header File

AVCalls.h

Related Methods

[AVAlert](#)
[AVDocAlertConfirm](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVDocClearSelection

```
ASBool AVDocClearSelection (AVDoc doc, ASBool highlight);
```

| | |
|------------------------|--|
| Description | Clears and destroys the current selection by calling the appropriate selection server's AVDocSelectionLosingSelectionProc . |
| Parameters | <p><i>doc</i></p> <p>The document whose selection is cleared.</p> <p><i>highlight</i></p> <p>Plug-ins should pass <i>true</i>, which tells the Acrobat viewer to remove the selection's highlighting; it has not already been removed. This only marks the highlighted regions of the display invalid, but does not immediately redraw the screen. Use AVPageViewDrawNow to force an immediate redraw if you wish.</p> |
| Return Value | <i>true</i> if the current selection was cleared, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | Broadcasts AVDocWillClearSelection if the selection type is not <i>ASAtomNull</i> . |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocSetSelection AVDocRegisterSelectionServer AVDocCopySelection AVDocEnumSelection AVDocDoSaveAsWithParams AVDocGetSelection AVDocDeleteSelection |

Example

```
PDTextSelect TextSelection;
HiliteEntry Hilite;
PDPage CurrentPDPage =
    PDDocAcquirePage(CurrentPDDoc, PageNum);
Hilite.offset= (ASUnsl6) Offset;
Hilite.length= 0;
TextSelection =
    PDTextSelectCreatePageHilite(CurrentPDPage,
    ,&Hilite, 1);
AVDocSetSelection(CurrentAVDoc,
    ASAtomFromString("Text"), (void *)
    TextSelection, true);
if(show)
    AVDocShowSelection(CurrentAVDoc);
else
    AVDocClearSelection(CurrentAVDoc, true);
PDPageRelease(CurrentPDPage);
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocClose

```
ASBool AVDocClose (AVDoc doc, ASBool noSave);
```

| | |
|------------------------|--|
| Description | <p>Closes the document window, optionally prompting the user to save the document if it has been modified. When this method closes the <i>AVDoc</i>, it also closes the underlying <i>PDDoc</i>.</p> <p>You can replace this method with your own version, using HFTReplaceEntry.</p> |
| Parameters | <p><i>doc</i></p> <p>The document to close.</p> <p><i>noSave</i></p> <p>If <i>true</i>, the document is closed without prompting the user and without saving, even if the document has been modified. Because this can cause data loss without user approval, use this feature judiciously.</p> <p>If <i>false</i>, prompts user to save the document if it has been modified.</p> |
| Return Value | <p><i>true</i> if the document closed, <i>false</i> if it did not (for example, if the user was prompted with the Save dialog and chose "Cancel"). The document will always close if <i>noSave</i> is <i>true</i>.</p> |
| Exceptions | None |
| Notifications | AVDocWillClose AVDocDidClose |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocOpenFromFile AVDocOpenFromFileWithParams AVDocOpenFromPDDoc AVDocOpenFromPDDocWithParams AVDocDoPrint |
| Example | <pre>AVDocClose(doc, true);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocCopyAction

```
PDAction AVDocCopyAction (AVDoc fromDoc, PDAction anAction,  
    AVDoc toDoc);
```

Description Makes a copy of the action for the (possibly different) specified [AVDoc](#). Calls the [AVActionCopyProc](#) callback in [AVActionHandlerProcs](#).

Note: The Acrobat viewers define [AVActionCopyProc](#) callbacks for all currently defined actions.

Parameters

fromDoc
The document whose action is copied.

anAction
The action to copy.

toDoc
The document to which the action is copied.

Return Value A copy of the action.

Exceptions Raises [avErrBadActionCopy](#) if there is no [AVActionCopyProc](#) registered in the action handler for *anAction*. Raises the standard exceptions if memory limit problems occur.

Notifications None

Header File *AVCalls.h*

Related methods [AVDocCopyActionCommon](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocCopyActionCommon

```
PDAction AVDocCopyActionCommon (AVDoc fromDoc,
    PDAction anAction, AVDoc toDoc);
```

| | |
|------------------------|---|
| Description | <p>Makes a copy of the action for the (possibly different) specified <i>AVDoc</i>.</p> <p>This copy includes only those fields that are described in the Portable Document Format Reference Manual as common to all actions. (This list currently includes Type, S, and Next.) This method is a proper “starting point” for the design of a Copy method for a custom action. This allows a plug-in to concentrate on those member objects specific to the custom action.</p> |
| Parameters | <p><i>fromDoc</i></p> <p>The document whose action is copied.</p> <p><i>anAction</i></p> <p>The action to copy.</p> <p><i>toDoc</i></p> <p>The document to which the action is copied.</p> |
| Return Value | A copy of the action. |
| Exceptions | Raises the standard exceptions if memory limit problems occur. |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related methods | AVDocCopyAction |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020003 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocCopyAdditionalActions

```
void AVDocCopyAdditionalActions (AVDoc doc, CosObj fromDict,
    AVDoc toDoc, CosObj toDict);
```

Description Copies any additional actions (**AA**) from a Cos Dictionary to a Cos Dictionary in the (possibly different) specified *AVDoc*.

This copy includes all those fields that are described in the [Portable Document Format Reference Manual](#) as common to all actions. (This list currently includes **E**, **X**, **D**, **U**, **O**, **C**, **FP**, **PP**, **NP**, and **LP**.) This method is designed to aid plug-ins in copying additional actions for pages.

For copying annotations, it is better to use [AVDocCopyAnnotCommon](#).

Parameters

doc
The document whose action is copied.

fromDict
The dictionary from which the action is copied.

toDoc
The document to which the action is copied.

toDict
The dictionary to which the action is copied.

Return Value A copy of the action.

Exceptions Raises the standard exceptions if memory limit problems occur.

Notifications None

Header File *AVCalls.h*

Related methods [AVDocCopyAnnot](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocCopyAnnot

```
PDAnnot AVDocCopyAnnot ( AVDoc fromDoc, PDAnnot anAnnot,
    AVDoc toDoc );
```

Description Makes a copy of the annotation for the (possibly different) specified [AVDoc](#). Calls the [AVAnnotHandlerCopyProc](#) callback in [AVAnnotHandler](#).

Note: The Acrobat viewers define [AVAnnotHandlerCopyProc](#) callbacks for all currently defined annotations.

Parameters

fromDoc
The document whose annotation is copied.

anAnnot
The annotation to copy.

toDoc
The document to which the annotation is copied.

Return Value A copy of the annotation.

Exceptions Raises [avErrBadAnnotationCopy](#) if there is no [AVAnnotHandlerCopyProc](#) registered in the annotation handler for *anAnnot*. Raises the standard exceptions if memory limit problems occur.

Notifications None

Header File [AVCalls.h](#)

Related methods [AVDocCopyAnnotCommon](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocCopyAnnotCommon

```
PDAnnot AVDocCopyAnnotCommon (AVDoc fromDoc, PDAnnot anAnnot,
AVDoc toDoc);
```

Description

Makes a copy of the annotation for the (possibly different) specified *AVDoc*.

This copy includes only those fields that are described in the [Portable Document Format Reference Manual](#) as common to all annotations. (This list currently includes **Type**, **Subtype**, **Rect**, **Border**, **C**, **T**, **M**, **F**, **H**, **AS**, **BS**, and **AA**.) This method is a proper “starting point” for the design of a Copy method for a custom annotation. This allows a plug-in to concentrate on those member objects specific to the custom annotation.

Parameters

fromDoc

The document whose annotation is copied.

anAnnot

The annotation to copy.

toDoc

The document to which the annotation is copied.

Return Value

A copy of the annotation.

Exceptions

Raises the standard exceptions if memory limit problems occur.

Notifications

None

Header File

AVCalls.h

Related methods

[AVDocCopyAnnot](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

Available in:

| | | |
|-------------------|-----------|------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

AVDocCopySelection

```
void AVDocCopySelection (AVDoc doc);
```

| | |
|------------------------|--|
| Description | <p>Copies the current selection to the clipboard, if possible. The selection is copied if the selection server has a AVDocSelectionCopyProc callback, and the selection server's AVDocSelectionCanCopyProc callback returns <i>true</i>. If the selection server does not have a AVDocSelectionCanCopyProc method, a default value of <i>true</i> is used.</p> <p>The selection is copied by calling the selection server's AVDocSelectionCopyProc callback.</p> |
| Parameters | <p><i>doc</i></p> <p>The document whose selection is copied.</p> |
| Return Value | None |
| Exceptions | Only those raised by the selection server's AVDocSelectionCopyProc and AVDocSelectionCanCopyProc callbacks. |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related methods | AVDocRegisterSelectionServer AVDocSetSelection AVDocDeleteSelection AVDocClearSelection AVDocGetSelectionType AVDocEnumSelection |

Example

```
PDTextSelect TextSelection;
HiliteEntry Hilite;
PDPage CurrentPDPage =
    PDDocAcquirePage(CurrentPDDoc, PageNum);
Hilite.offset = (ASUnsl6) Offset;
Hilite.length = 0;
TextSelection =
    PDTextSelectCreatePageHilite(
        CurrentPDPage, &Hilite, 1);
AVDocSetSelection(CurrentAVDoc,
    ASAtomFromString("Text"), (void *)
    TextSelection, true);
AVDocCopySelection(doc);
AVDocDeleteSelection(doc);
PDPageRelease(CurrentPDPage);
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocDeleteSelection

```
ASBool AVDocDeleteSelection (AVDoc doc);
```

Description Deletes the specified document's current selection, if possible. The selection is deleted if changing the selection is currently permitted, the selection server has an [AVDocSelectionDeleteProc](#) callback, and the selection server's [AVDocSelectionCanDeleteProc](#) callback returns *true*. If the selection server does not have a [AVDocSelectionCanDeleteProc](#) callback, a default value of *true* is used.

The selection is deleted by calling the selection server's [AVDocSelectionDeleteProc](#) callback.

Parameters *doc*

The document whose selection is deleted.

Return Value *true* if the current selection was actually deleted, *false* otherwise.

Exceptions Only those raised by the selection server's [AVDocSelectionDeleteProc](#) and [AVDocSelectionCanDeleteProc](#) callbacks.

Notifications None

Header File *AVCalls.h*

Related methods [AVDocRegisterSelectionServer](#)
[AVDocGetSelection](#)
[AVDocSetSelection](#)
[AVDocClearSelection](#)
[AVDocCopySelection](#)
[AVDocEnumSelection](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

AVDocDoActionPropsDialog

```
ASBool AVDocDoActionPropsDialog (AVDoc doc, PDAction* action,
    const char* dialogTitle);
```

Description Displays a modal dialog that allows a user to specify an action. Used, for example, by forms to add actions to fields.

Parameters

doc
The document in which the action is specified.

action
The specified action.

dialogTitle
Title for the dialog.

Return Value *true* if the user clicked the “Set Link” button, *false* if the user clicked “Cancel.”

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVDocPerformAction](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

AVDocDoCopyAs

```
ASBool AVDocDoCopyAs (AVDoc doc);
```

Description Prompts the user with a standard file dialog and copies the file *doc* byte for byte. Displays a progress monitor showing the progress of the file copy.

Parameters *doc*
The document to copy.

Return Value *true* if the copy was successful, *false* otherwise.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVDocDoSave](#)
[AVDocDoSaveAs](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocDoPrint

```
void AVDocDoPrint (AVDoc doc);
```

| | |
|------------------------|--|
| Description | Performs the print operation, including user dialogs. You can replace this method with your own version, using HFTReplaceEntry . |
| Parameters | <i>doc</i> The document to print. |
| Return Value | None |
| Notifications | AVDocDidPrint AVDocWillPrint |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocPrintPages AVDocPrintPagesWithParams |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVDocDoSave

```
ASBool AVDocDoSave (AVDoc doc);
```

Description

Saves a file, including handling any user interface (for example, “Save File” dialog) that is needed. Plug-ins do not need to call this method directly, but it is available for plug-ins that need to override the Acrobat viewer’s built-in save method, for example to save the changes made to a PDF file into a special file, but not save the original PDF file.

You can replace this method with your own version, using [HFTReplaceEntry](#).

Parameters

doc

The document to save.

Return Value

true if the document was successfully saved, *false* otherwise.

Exceptions

None

Notifications

None

Header File

AVCalls.h

Related Methods

[AVDocClose](#)
[AVDocDoSaveAs](#)
[AVDocDoSaveAsWithParams](#)
[AVDocOpenFromFile](#)
[AVDocOpenFromFileWithParams](#)
[AVDocOpenFromPDDoc](#)
[AVDocOpenFromPDDocWithParams](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

AVDocDoSaveAs

```
ASBool AVDocDoSaveAs (AVDoc doc);
```

| | |
|------------------------|---|
| Description | Displays a file dialog and saves the document to a potentially new name. Allows plug-ins (such as Optimizer) to do their own file saving. You can replace this method with your own version, using HFTReplaceEntry . |
| Parameters | <i>doc</i> The document to save. |
| Return Value | <i>true</i> if the document was successfully saved, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocDoPrint AVDocDoSave AVDocDoSaveAsWithParams |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----------|------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

AVDocDoSaveAsWithParams

```
ASBool AVDocDoSaveAsWithParams (AVDoc doc,  
    AVDocSaveParams params);
```

| | |
|------------------------|---|
| Description | Saves a file, using the parameters specified in <i>paramsIn</i> . You can replace this method with your own version, using HFTReplaceEntry . |
| Parameters | <i>doc</i> The document to save. <i>params</i> A structure with information telling how the <i>AVDoc</i> is saved. |
| Return Value | <i>true</i> if the document was successfully saved, <i>false</i> otherwise. |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocDoSave AVDocDoSaveAs |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVDocDoSelectionProperties

```
void AVDocDoSelectionProperties (AVDoc doc);
```

Description Displays the user interface, if any, for setting the current selection's properties. It does this by invoking the [AVDocSelectionPropertiesProc](#) callback, if any, of the current selection's selection server.

Parameters *doc*
The document containing the selection whose properties are set.

Return Value None

Exceptions Only those raised by the selection server's [AVDocSelectionPropertiesProc](#) callback.

Notifications None

Header File *AVCalls.h*

Related Methods [AVDocRegisterSelectionServer](#)
[AVDocSetSelection](#)

Example

```
if(PtInRect(&myRect, point))  
    AVDocDoSelectionProperties(doc);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocEnumSelection

```
void AVDocEnumSelection (AVDoc doc, AVSelectionEnumProc proc,  
void* clientData);
```

| | |
|------------------------|--|
| Description | Enumerates the elements of the current selection by calling the current selection server's AVDocSelectionEnumSelectionProc callback. If the selection server does not have an AVDocSelectionEnumSelectionProc , calls <i>proc</i> and passes the entire selection to it in the <i>aSelectedObject</i> parameter. |
| Parameters | <p><i>doc</i></p> <p>The document whose selection is enumerated.</p> <p><i>proc</i></p> <p>User-supplied callback to call for each element in the selection.</p> <p>Enumeration ends if <i>proc</i> returns <i>false</i>.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>proc</i> each time it is called.</p> |
| Return Value | None |
| Exceptions | Only those raised by the selection server's AVDocSelectionEnumSelectionProc callback, and those raised by <i>proc</i> . |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocRegisterSelectionServer AVDocSetSelection AVDocClearSelection AVDocDeleteSelection AVDocCopySelection AVDocSelectionEnumPageRanges |

Example

```
static ACCB1 ASBool ACCB2 mySelectEnum(
    AVDoc doc, void* clientData,
    void* selectedObject)
{
    ASInt32 page = PDTextSelectGetPage(
        (PDTextSelect) selectedObject);
    return false;
}

AVDocEnumSelection(doc,
    ASCallbackCreateProto(
        AVSelectionEnumProc, &mySelectEnum),
    NULL);
```

Availability

Plug-ins: Available if *PL_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocGetAVWindow

[AVWindow](#) AVDocGetAVWindow ([AVDoc](#) doc);

| | |
|------------------------|---|
| Description | Gets the <i>AVWindow</i> in which the document is being displayed. |
| Parameters | <i>doc</i> The document whose <i>AVWindow</i> is obtained. |
| Return Value | The document's <i>AVWindow</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocGetPageView AVDocGetViewMode |
| Example | <pre>AVWindow myWin = AVDocGetAVWindow(doc);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocGetClientName

```
ASInt32 AVDocGetClientName (AVDoc avDoc, char* buffer,
    ASInt32 bufSize);
```

| | |
|------------------------|--|
| Description | Gets the <i>AVDoc</i> client (container application) name. This method can be used by a plug-in to determine which client application caused the Acrobat viewer to open a document. |
| Parameters | <p><i>avDoc</i> The document whose client name is obtained.</p> <p><i>buffer</i> (Filled by the method) The buffer into which the client name is written. May be up to 255 characters. The client name is <i>NULL</i>-terminated. If the client name is longer than <i>bufSize</i>, the first <i>bufSize</i> – 1 bytes are copied into it, followed by a <i>NULL</i>.</p> <p><i>bufSize</i> The size of <i>buffer</i>, in bytes.</p> |
| Return Value | The number of characters written into <i>buffer</i> , excluding the <i>NULL</i> termination. Returns 0 if the specified document does not have a client associated with it. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocSetClientName |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020001 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocGetPageText

```
void AVDocGetPageText (AVDoc doc, ASInt32 pageNum,  
    PDTextSelect pdText, ASAtom format, AVTextCopyProc copyProc,  
    void* clientData);
```

Description Gets the text from the specified text selection, converts it to the specified format, and passes it to a user-supplied procedure.

Parameters *doc*

The document from which text is obtained.

pageNum

The page number in *doc* from which text is obtained. The first page in a document is page 0.

pdText

The text selection whose text is obtained. Passes *NULL* to get all the text on the page.

format

The format in which text is desired. The following are allowed values (these strings must be converted to *ASAtoms* using [ASAtomFromString](#)):

All — (Macintosh/Windows) Calls *copyProc* once with each available format.

Text — (Macintosh/Windows) Calls *copyProc* twice. The first time, it passes the text in the specified selection. The text is passed using the same platform encoding as when it is put onto the clipboard (*format* = *Text*). The second time, it passes a Unicode version of the text (*format* = *UCSText*).

Style — (Macintosh only) Calls *copyProc* twice. The first time, it passes the *Text* representation. The second time, it passes a *StyleInfo* record.

RTF — (Macintosh/Windows) Calls *copyProc* twice. The first time, it passes the text in Rich Text Format. The second time, it passes a Unicode version of the text.

Upon using/manipulating these texts, you should check the format of these texts in *copyProc*.

copyProc

User-supplied callback to which the text is passed.

clientData

Pointer to user-supplied data to pass to *copyProc* each time it is called.

Return Value None

Exceptions [pdErrOpNotPermitted](#)

Notifications None

Header File *AVCalls.h*

Related Methods [PDTextSelectCreateRanges](#)
[PDTextSelectCreatePageHilite](#)
[PDTextSelectCreateWordHilite](#)
[PDDocCreateStructTreeRoot](#)
[ASAtomFromString](#)
[PDWordFinderEnumWords](#)
[PDWordFinderAcquireWordList](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020001 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocGetPageView

[AVPageView](#) AVDocGetPageView ([AVDoc](#) doc);

Description Gets the *AVPageView* for the specified document.
If the *AVDoc* is embedded in an external window (such as an embedded PDF in HTML in a Web browser's window), returns *NULL*.

Parameters *doc*
The document whose *AVPageView* is obtained.

Return Value The document's *AVPageView*. *NULL* if the *AVDoc* is embedded in an external window.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVDocGetAVWindow](#)
[AVDocGetViewMode](#)
[AVDocIsExternal](#)
[AVPageViewDrawRect](#)
[PDFPageDrawContentsToWindow](#)

Example
AVPageView avPageView =
AVDocGetPageView(AVAppGetActiveDoc());

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocGetPDDoc

[PDDoc](#) AVDocGetPDDoc ([AVDoc](#) avDoc);

Description Gets the *PDDoc* to associate with the specified *AVDoc*.

Parameters *avDoc*
The document whose *PDDoc* is obtained.

Return Value The *PDDoc* associated with *avDoc*.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppEnumDocs](#)
[AVDocOpenFromPDDoc](#)
[PDDocImportNotes](#)

Example

```
ACCB1 ASBool ACCB2 myAppEnumProc(
    AVDoc doc, void* clientData)
{
    if (PDDocGetNumPages (AVDocGetPDDoc (doc) )
        > 100)
        AAlertNote ("Long document");
    return true;
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocGetSelection

```
void* AVDocGetSelection (AVDoc doc);
```

| | |
|------------------------|---|
| Description | Gets the current selection for the specified document. |
| Parameters | <p><i>doc</i></p> <p>The document whose selection is obtained.</p> |
| Return Value | The current selection, or <i>NULL</i> if there is no selection. See Selection Types for a list of the data types returned for the built-in selection types. A <i>NULL</i> return value from this method is not sufficient to determine that there is no selection, use AVDocGetSelectionType instead. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocSetSelection AVDocGetSelectionType AVDocClearSelection AVDocDeleteSelection AVDocEnumSelection AVDocCopySelection |

Example

```
AVGrafSelect gs;

if(ASAtomFromString("Bitmap") ==
    AVDocGetSelectionType(doc)) {
    gs =
        (AVGrafSelect)AVDocGetSelection(doc);
}
```

| | |
|---------------------|---|
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
|---------------------|---|

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocGetSelectionServerByType

[AVDocSelectionServer](#) AVDocGetSelectionServerByType ([ASAtom](#) type);

| | |
|------------------------|--|
| Description | Gets the selection server that handles the specified selection type. |
| Parameters | <i>type</i> The <i>ASAtom</i> corresponding to the type for which the selection server was registered. The string for <i>type</i> can be converted to an <i>ASAtom</i> using ASAtomFromString . |
| Return Value | The selection server that handles the specified type. Returns <i>NULL</i> if no such selection server is registered. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocGetSelectionServerByType ASAtomFromString |
| Example | <pre>AVDocSelectionServer server; if(server = AVDocGetSelectionServerByType(ASAtomFromString("imageSelect")) { AVDocClearSelection(doc, true); }</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocGetSelectionType

[ASAtom](#) AVDocGetSelectionType ([AVDoc](#) doc);

| | |
|------------------------|--|
| Description | Gets the current selection's type for the specified document. |
| Parameters | <p><i>doc</i></p> <p>The document whose selection type is obtained.</p> |
| Return Value | The <i>ASAtom</i> corresponding to the current selection type. Returns <i>ASAtomNull</i> if there is no selection. The <i>ASAtom</i> returned can be converted to a string using ASAtomGetString . See Selection Types for a list of the built-in selection types. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocSetSelection ASAtomGetString AVDocGetSelection |
| Example | <pre>PDTextSelect ts; if(ASAtomFromString("Text") == AVDocGetSelectionType(doc)) { ts = AVDocGetSelection(doc); }</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocGetSplitterPosition

```
ASInt16 AVDocGetSplitterPosition (AVDoc doc);
```

Description Gets the splitter position. The splitter is the vertical division between the bookmark/thumbnail pane and the document pane. The default splitter location is saved in the Acrobat viewer's preferences file, and can be read/set using [AVAppGetPreference](#) and [AVAppSetPreference](#).

Parameters *doc*
The document whose splitter position is obtained.

Return Value The width of the bookmark/thumbnail pane, measured in pixels. Returns 0 if the bookmark/thumbnail pane is not currently displayed.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVDocSetSplitterPosition](#)
[AVAppGetPreference](#)

Example

```
if (AVDocGetSplitterPosition(doc) < 72)
    AVDocSetSplitterPosition(doc, 72);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocGetViewDef

```
void AVDocGetViewDef (AVDoc doc, AVDocViewDef viewDef);
```

Description Fills out the given [AVDocViewDef](#) structure with the information needed to restore this document's state at a later date. You must set the "use" fields as desired.

Parameters

doc
The document whose state is recorded.

viewDef
A pointer to the structure that stores the state.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVDocSetViewDef](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocGetViewMode

[PDPageMode](#) AVDocGetViewMode ([AVDoc](#) doc);

| | |
|------------------------|---|
| Description | Gets the current view mode. |
| Parameters | <p><i>doc</i></p> <p>The document whose view mode is obtained.</p> |
| Return Value | The current view mode. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocSetViewMode PDDocGetPageLabel |
| Example | <pre>if (AVDocGetViewMode(doc) == PDUseNone) AVDocSetViewMode(doc, PDUseBookmarks);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocIsExternal

```
ASBool AVDocIsExternal (AVDoc doc);
```

| | |
|------------------------|---|
| Description | Determines whether a given document is being displayed in an application's external window (such as in a Web browser's window). |
| Parameters | <p><i>doc</i></p> <p>The document being tested.</p> |
| Return Value | <i>true</i> if the given document is being displayed in an application's external window, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocGetPageView |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocIsReadOnly

```
ASBool AVDocIsReadOnly (AVDoc doc);
```

| | |
|------------------------|---|
| Description | Checks to see if an <i>AVDoc</i> is read-only. |
| Parameters | <p><i>doc</i></p> <p>The <i>AVDoc</i> whose read-only state is checked.</p> |
| Return Value | <i>true</i> if the given document is read-only, <i>false</i> otherwise. |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocSetReadOnly |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVDocOpenFromASFileWithParams

```
AVDoc AVDocOpenFromASFileWithParams (ASFile file,  
char* tempTitle, AVDocOpenParams params);
```

Description Opens and displays a document from an *ASFile*, using the specified parameters to control the window's size, location, and visibility.

If you want to display a page from a PDF file as a bitmap, use [PDPPageDrawContentsToWindow](#).

Parameters

file
The *ASFile* to open.

tempTitle
An optional window title for the document.

params
Open parameters for the document.

Return Value An *AVDoc* object for *file*.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVDocOpenFromFileWithParams](#)
[AVDocOpenFromPDDocWithParams](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocOpenFromFile

```
AVDoc AVDocOpenFromFile (ASPathName pathName,  
    ASFileSys fileSys, char* tempTitle);
```

| | |
|----------------------|--|
| Description | <p>Opens and displays a document from a file. This is equivalent to</p> <p><u>AVDocOpenFromFileWithParams</u>(<i>pathName</i>, <i>fileSys</i>, <i>tempTitle</i>, <i>NULL</i>);</p> <p>If you want to display a page from a PDF file as a bitmap, use <u>PDFPageDrawContentsToWindow</u>.</p> <p><i>Note: Do not open and then immediately close an AVDoc without letting at least one event loop complete.</i></p> |
| Parameters | <p><i>pathName</i></p> <p>The file to open.</p> <p><i>fileSys</i></p> <p>The file system on which <i>pathName</i> resides.</p> <p><i>tempTitle</i></p> <p>If <i>tempTitle</i>!=<i>NULL</i>, <i>pathname</i> is a temporary file, and <i>tempTitle</i> is used as the window's title.</p> |
| Return Value | The document that was opened. Returns <i>NULL</i> if the viewer failed to open the document. |
| Exceptions | None |
| Notifications | <p><u>AVDocWillOpenFromFile</u></p> <p><u>AVDocDidOpen</u></p> <p><u>AVAppFrontDocDidChange</u></p> <p><u>AVDocDidActivate</u></p> <p><u>AVDocDidDeactivate</u></p> <p>The following notifications are broadcast if the document has a valid open action:</p> <p><u>AVDocWillPerformAction</u></p> <p><u>AVDocDidPerformAction</u></p> |
| Header File | <i>AVCalls.h</i> |

Related Methods [AVDocOpenFromFileWithParams](#)
[AVDocOpenFromPDDoc](#)
[AVDocOpenFromPDDocWithParams](#)
[AVDocClose](#)
[ASGetDefaultFileSys](#)

Example

```
AVDoc myDoc = AVDocOpenFromFile(
    myPathName, ASGetDefaultFileSys(),
    NULL);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocOpenFromFileWithParams

```
AVDoc AVDocOpenFromFileWithParams (ASPathName pathName,  
    ASFileSys fileSys, char* tempTitle, AVDocOpenParams params);
```

Description

Opens and displays a document from a file, using the specified parameters to control the window's size, location, and visibility.

You can replace this method with your own version, using [HFTReplaceEntry](#).

If you want to display a page from a PDF file as a bitmap, use [PDFPageDrawContentsToWindow](#).

Note: Do not open and then immediately close an AVDoc without letting at least one event loop complete.

Parameters

pathName

The file to open.

fileSys

The file system on which *pathName* resides.

tempTitle

If *tempTitle*!=NULL, *pathName* is a temporary file and *tempTitle* is used as the window's title.

params

Parameters used when opening the file. Can be NULL.

Return Value

The document that was opened.

Exceptions

None

Notifications

[AVDocWillOpenFromFile](#)

[AVDocDidOpen](#)

[AVAppFrontDocDidChange](#)

[AVDocDidActivate](#)

[AVDocDidDeactivate](#)

The following notifications are broadcast if the document has a valid open action:

[AVDocWillPerformAction](#)

[AVDocDidPerformAction](#)

Header File

AVCalls.h

Related Methods [AVDocIsReadOnly](#)
[AVDocOpenFromFile](#)
[AVDocOpenFromPDDoc](#)
[AVDocOpenFromPDDocWithParams](#)
[AVDocClose](#)
[ASGetDefaultFileSys](#)

Example

```
AVDocOpenParamsRec params;
AVDoc myDoc;

params.size = sizeof(AVDocOpenParamsRec);
params.useFrame = false;
params.useVisible = true;
params.visible = false;
params.useServerType = false;
params.useReadOnly = true;
params.readOnly = true;
params.useViewType = true;
params.viewType = "AVPageView";
myDoc =
    AVDocOpenFromFileWithParams(myPathName,
                                NULL, NULL, &params);
```

Availability Plug-ins: Available if *PL_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocOpenFromPDDoc

```
AVDoc AVDocOpenFromPDDoc (PDDoc doc, char* tempTitle);
```

Description

Opens and returns an *AVDoc* view of *pdDoc*. In addition, acquires *pdDoc*. This method is equivalent to [AVDocOpenFromPDDocWithParams](#)(*pdDoc*, *tempTitle*, *NULL*);

If you want to display a page from a PDF file as a bitmap, use [PDFPageDrawContentsToWindow](#).

Note: Do not open and then immediately close an AVDoc without letting at least one event loop complete.

*Note: [AVDocClose](#) should be used in place of [PDDocClose](#) after AVDocOpenFromPDDoc is called. [AVDocClose](#) will decrement the *pdDoc* appropriately and free document-related resources.*

Parameters

doc

The document to open.

tempTitle

If *tempTitle*!=*NULL*, *pathname* is a temporary file and *tempTitle* is used as the window's title.

Return Value

NULL if failure occurs.

Exceptions

Raises [genErrGeneral](#) if *pdDoc* is *NULL* or has 0 pages. Raises [pdErrBadAction](#) if the document's open action is recursive. Raises [avErrCantOpenMoreThanTenDocs](#) if the maximum number of documents is already open. Raises [genErrNoMemory](#) if there is insufficient memory to open the document.

Notifications

[AVAppFrontDocDidChange](#)
[AVDocDidActivate](#)
[AVDocDidDeactivate](#)

The following notifications are broadcast if the document has a valid open action:

[AVDocWillPerformAction](#)
[AVDocDidPerformAction](#)

Header File

AVCalls.h

Related Methods [AVDocOpenFromPDDocWithParams](#)
[AVDocOpenFromFile](#)
[AVDocOpenFromFileWithParams](#)
[AVDocClose](#)

Example `AVDocOpenFromPDDoc(somePDDoc , NULL) ;`

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocOpenFromPDDocWithParams

```
AVDoc AVDocOpenFromPDDocWithParams (PDDoc pdDoc,  
char* tempTitle, AVDocOpenParams params);
```

| | |
|------------------------|--|
| Description | <p>Opens and displays a document from a <i>PDDoc</i>, using the specified parameters to control the window's size, location, and visibility.</p> <p>If you want to display a page from a PDF file as a bitmap, use PDPageDrawContentsToWindow.</p> <p><i>Note: Do not open and then immediately close an AVDoc without letting at least one event loop complete.</i></p> |
| Parameters | <p><i>pdDoc</i></p> <p>The document to open and display.</p> <p><i>tempTitle</i></p> <p>If <i>tempTitle</i>!=NULL, <i>pathname</i> is a temporary file and <i>tempTitle</i> is used as the window's title.</p> <p><i>params</i></p> <p>Parameters used when opening the file. Can be NULL.</p> |
| Return Value | The document that was opened. |
| Exceptions | None |
| Notifications | <p>AVAppFrontDocDidChange</p> <p>AVDocDidActivate</p> <p>AVDocDidDeactivate</p> <p>The following notifications are broadcast if the document has a valid open action:</p> <p>AVDocWillPerformAction</p> <p>AVDocDidPerformAction</p> |
| Header File | <i>AVCalls.h</i> |
| Related Methods | <p>AVDocIsReadOnly</p> <p>AVDocOpenFromFile</p> <p>AVDocOpenFromFileWithParams</p> <p>AVDocOpenFromPDDoc</p> <p>AVDocClose</p> |

Example

```
AVDocOpenParamsRec params;

params.size = sizeof(AVDocOpenParamsRec);
params.useFrame = false;
params.useVisible = true;
params.visible = false;
AVDocOpenFromPDDocWithParams(somePDDoc,
    NULL, &params);
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocPerformAction

```
void AVDocPerformAction (AVDoc doc, PDAction action);
```

| | |
|------------------------|---|
| Description | Performs an action. |
| Parameters | <p><i>doc</i></p> <p>The document containing the action to perform.</p> <p><i>action</i></p> <p>The action to perform.</p> |
| Return Value | None |
| Exceptions | pdErrBadAction |
| Notifications | AVDocWillPerformAction AVDocDidPerformAction |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocDoActionPropsDialog PDBookmarkGetAction PDDocGetOpenAction PDLinkAnnotGetAction |

Example

```

DURING
    action = PDBookmarkGetAction(item);
HANDLER
    err = ERRORCODE;
END_HANDLER
if(!err){
    DURING
        AVDocPerformAction(doc, action);
    HANDLER
        err = ERRORCODE;
    END_HANDLER
}
    
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocPrintPages

```
void AVDocPrintPages (AVDoc doc, ASInt32 firstPage,  
    ASInt32 lastPage, ASInt32 psLevel, ASBool binaryOK,  
    ASBool shrinkToFit);
```

Description

Prints without displaying any user dialogs. The current printer, page settings, and job settings are used. Printing is complete when this method returns.

You can replace this method with your own version, using [HFTReplaceEntry](#).

Note: If security has been set on a file so that it is not printable, the document won't print, but no error is raised. Check the security before printing the file.

Parameters

doc

The document from which pages are printed.

firstPage

The first page in *doc* to print.

lastPage

The last page in *doc* to print.

psLevel

Applies to PostScript printing. Must be either 1 or 2. If 1, Level 1 PostScript code is generated. If 2, Level 2 PostScript code is generated.

binaryOK

Applies to PostScript printing. If *true*, the PostScript code may contain binary data. If *false*, all binary data is encoded into an ASCII format.

shrinkToFit

If *true*, the page is shrunk (if necessary) to fit into the imageable area of a page in the printer. If *false*, pages are printed at actual size and may appear clipped on the printed page.

Return Value

None

Exceptions Raises [genErrBadParm](#) if an invalid parameter is provided. Can raise any of the CosErrExpected exceptions, such as [cosErrExpectedDirect](#) or [cosErrExpectedNumber](#).

In general, this method can raise any exception that can occur during the parsing of a page and its resources, such as [pdErrUnknownProcsets](#) or [pdErrUnableToExtractFontErr](#).

Notifications [PDDocWillPrintPages](#)
[PDDocWillPrintPage](#)
[PDDocDidPrintPage](#)
[PDDocDidPrintPages](#)

Header File *AVCalls.h*

Related Methods [AVDocDoPrint](#)
[AVDocPrintPagesWithParams](#)

Example

```
AVDocPrintPages(AVAppGetActiveDoc(), 0, 2,
                2, false, false);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocPrintPagesWithParams

```
void AVDocPrintPagesWithParams (AVDoc doc,  
    AVDocPrintParams params);
```

| | |
|------------------------|---|
| Description | <p>Prints a document with a full range of options. Printing is complete when this method returns.</p> <p>Allows interactive printing to the specified printer (on most platforms).</p> <p>Performs “embedded” printing. That is, allows a PDF page to print within a bounding rectangle on a page. You can replace this method with your own version, using HFTReplaceEntry.</p> <p><i>Note: If security has been set on a file so that it is not printable, the document won't print, but no error is raised. Check the security before printing the file.</i></p> |
| Parameters | <p><i>doc</i></p> <p>The <i>AVDoc</i> from which to print pages.</p> <p><i>params</i></p> <p>See AVDocPrintParams.</p> <p><i>Note: With the Reader, you cannot use the emitToFile flag. For the Reader, use the emitToPrinter, interactive, or embedded flags.</i></p> |
| Return Value | None |
| Exceptions | <p>Raises genErrBadParm if an invalid parameter is provided. Can raise any of the CosErrExpected exceptions, such as cosErrExpectedDirect or cosErrExpectedNumber.</p> <p>In general, this method can raise any exception that can occur during the parsing of a page and its resources, such as pdErrUnknownProcsets or pdErrUnableToExtractFontErr.</p> |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocDoPrint AVDocPrintPages |

Example

```
AVDocPrintParamsRec myParams;
AVDoc doc;
char buf[255];
ASPathName name = NULL;
doc = AVAppGetActiveDoc();

name = ASPathFromPlatformPath(
    (void*)"c:\\test.ps");

if (name) {
    AAlertNote("Printing Starts");
    memset(&myParams, 0,
        sizeof(AVDocPrintParamsRec));
    myParams.size =
        sizeof(AVDocPrintParamsRec);

    myParams.emitToFile = true;
    myParams.interactive = false;
    myParams.emitToPrinter = false;
    myParams.embedded = false;

    myParams.doColorSeparations = false;
    myParams.binaryOK = false;
    myParams.shrinkToFit = true;
    myParams.firstPage = -1L; /* should print
        all pages */
    myParams.lastPage = -1L;
    myParams.psLevel = 2L;

    myParams.fileSysName = ASAtomNull;
    myParams.filePathName = name;
    myParams.printerSpec = NULL;

    myParams.doColorSeparations = false;
    myParams.emitFileOption = kAVEmitFilePS;
    myParams.emitFontOption =
        kAVEmitFontEmbeddedFonts;
DURING
    AVDocPrintPagesWithParams(doc,
        &myParams);

HANDLER
    ASGetErrorString(ERRORCODE, buf,
        sizeof(buf));
    AAlertNote(buf);
    return;
```

```

END_HANDLER
    AAlertNote("Printing Ends");
}
else
    AAlertNote("bad path name");

```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocRegisterSelectionServer

```
ASBool AVDocRegisterSelectionServer
(AVDocSelectionServer server);
```

| | |
|------------------------|--|
| Description | <p>Registers a new selection server with the Acrobat viewer. Selection servers allow the selection of items other than those that can be selected in the as-shipped Acrobat viewer. For example, a selection server could allow a user to select a sampled image.</p> <p>This method can be used to replace an existing selection server that handles the same selection type.</p> |
| Parameters | <p><i>server</i></p> <p>Structure containing the selection server's callback functions. This structure must <i>not</i> be freed after calling AVDocRegisterSelectionServer.</p> |
| Return Value | Always returns <i>true</i> . |
| Exceptions | Raises genErrBadParm if server is <i>NULL</i> , if the <i>size</i> field in the <i>AVDocSelectionServerRec</i> is incorrect, or if the selection server's AVDocSelectionGetTypeProc is <i>NULL</i> . |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocClearSelection AVDocCopySelection AVDocDeleteSelection AVDocDoSaveAsWithParams AVDocEnumSelection AVDocGetSelection AVDocShowSelection |

Example

```
AVDocRegisterSelectionServer:
static AVDocSelectionServerRec mySelServer;

DrawImageSelectionCallback =
    ASCallbackCreateProto(
        AVPageViewDrawProc,
        &DrawImageSelection);
memset(&mySelServer, 0,
    sizeof(AVDocSelectionServerRec));
mySelServer.size =
    sizeof(AVDocSelectionServerRec); /*
    important for future */
mySelServer.GetType =
    ASCallbackCreateProto(
        AVDocSelectionGetTypeProc,
        &mySelServerGetType);
mySelServer.GettingSelection =
    ASCallbackCreateProto(
        AVDocSelectionGettingSelectionProc,
        &mySelServerGettingSelection);
mySelServer.LosingSelection =
    ASCallbackCreateProto(
        AVDocSelectionLosingSelectionProc,
        &mySelServerLosingSelection);
mySelServer.ShowSelection =
    ASCallbackCreateProto(
        AVDocSelectionShowSelectionProc,
        &mySelServerShowSelection);

AVDocRegisterSelectionServer(&mySelServer)
;
```

Availability

Plug-ins: Available if *PL_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocSelectionEnumPageRanges

```
void AVDocSelectionEnumPageRanges (AVDoc doc,  
    AVSelectionPageRangeEnumProc enumProc, void* clientData);
```

| | |
|------------------------|--|
| Description | <p>Enumerates the pages on which there is a selection by calling the current selection server's AVDocSelectionEnumPageRangesProc callback.</p> <p>This method allows determining which pages are currently involved in a selection, regardless of the selection type. This facilitates discovering whether a given point is in a selection or not.</p> <p>Pages are enumerated in ascending order, and consecutive pages are grouped into a single page range.</p> |
| Parameters | <p><i>doc</i></p> <p>The <i>AVDoc</i> from which to enumerate page ranges.</p> <p><i>enumProc</i></p> <p>User-supplied function that is called for each page on which there is a selection.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>enumProc</i> each time it is called.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocEnumSelection |

Example

```
AVDoc avDoc;
AVSelectionPageRangeEnumProc
    pageSelectionEnumeratorCB;

pageSelectionEnumeratorCB =
    ASCallbackCreateProto
        (AVSelectionPageRangeEnumProc,
         pageSelectionEnumerator);
AVDocSelectionEnumPageRanges(avDoc,
    pageSelectionEnumeratorCB, void);

ACCB1 ASBool ACCB2 pageSelectionEnumerator
    (AVDoc doc, void* clientData,
     ASInt32 firstPage, ASInt32 lastPage) {
    /* Some operations on each page range */

}
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVDocSendAuxData

```
ASBool AVDocSendAuxData (AVDoc avDoc, ASAtom auxDataType,
    void* auxData, ASInt32 auxDataLen);
```

Description Sends auxiliary data to an *AVDoc*. If an *AuxDataHandler* exists for the data type, the handler is called with *avDoc*, *auxDataType*, and *auxData*. The definition of the auxiliary data is dependent on the *AuxDataHandler*. For any value of *auxDataType*, the *auxData* parameter must be predefined so that a user of this method knows what type of data to send. It is expected that the implementor of an *AuxDataHandler* provides this definition.

Parameters

avDoc
The target document.

auxDataType
The type of data being sent.

auxData
A pointer to the data.

auxDataLen
The length of the data.

Return Value *true* if *auxData* was accepted by a handler, *false* otherwise.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVHasAuxDataHandler](#)
[AVRegisterAuxDataHandler](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocSetClientName

```
void AVDocSetClientName (AVDoc avDoc, char* clientName);
```

Description Sets the *AVDoc* client (container application) name. This method can be used by plug-ins that open documents in the viewer via DDE or Apple events. Most plug-ins will not open documents in this way, however, making this method unnecessary for most plug-ins.

Parameters

avDoc

The document whose client names is set.

clientName

The buffer from which the client name is read. May be up to 255 characters, and must be *NULL*-terminated.

Return Value None

Exceptions [genErrNoMemory](#)

Notifications None

Header File *AVCalls.h*

Related Methods [AVDocGetClientName](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020001 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocSetDead

```
void AVDocSetDead (AVDoc doc, ASBool dead);
```

Description

Indicates the file stream for this document is terminated, although the *AVDoc* is still open. No more data can be read from this *AVDoc*.

AVDocs that are marked as dead may start returning *NULL* at anytime between the initial call to *AVDocSetDead* and the time that the *AVDoc* is actually closed. So callers of [AVDocGetAVWindow](#) and other [AVDoc](#) property methods must check for a *NULL* return value.

Parameters

doc

The document to set as dead.

dead

true if the document's file stream is terminated, *false* otherwise.

Return Value

None

Exceptions

None

Notifications

If *dead* is *true*, broadcasts [AVDocWantsToDie](#).

Header File

AVCalls.h

Related Methods

[AVWindowGetPlatformThing](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocSetReadOnly

```
void AVDocSetReadOnly (AVDoc doc, ASBool readOnly);
```

| | |
|------------------------|---|
| Description | Sets the read-only state of an <i>AVDoc</i> . |
| Parameters | <i>doc</i> The <i>AVDoc</i> to set to read-only. <i>readOnly</i> <i>true</i> if the given document is set to read-only, <i>false</i> if it is set to read-write. |
| Return Value | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocIsReadOnly |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVDocSetSelection

```
ASBool AVDocSetSelection (AVDoc doc, ASAtom type, void* data,
    ASBool highlight);
```

| | |
|---------------------|---|
| Description | Sets the document's current selection to the specified selection by calling the appropriate selection server's AVDocSelectionGettingSelectionProc callback. Clears the previous selection, if any, by calling the previous selection server's AVDocSelectionLosingSelectionProc callback. |
| Parameters | <p><i>doc</i></p> <p>The <i>AVDoc</i> in which the selection is set.</p> <p><i>type</i></p> <p>Selection type. Can be either a built-in type or one supported by a selection server added by a plug-in. Can be converted to an <i>ASAtom</i> using ASAtomFromString. See Selection Types for a list of the built-in selection types.</p> <p><i>data</i></p> <p>Data structure representing the selection. <i>data</i>'s type depends on what is passed in <i>type</i>. See Selection Types for a list of the data types for the built-in selection types.</p> <p><i>highlight</i></p> <p>Plug-ins should pass <i>true</i>, which tells the Acrobat viewer to highlight the selection because it has not already been highlighted. This only marks the highlighted regions of the display invalid, but does not immediately redraw the screen. Use AVPageViewDrawNow to force an immediate redraw if you wish.</p> |
| Return Value | <p><i>true</i> if the selection was set successfully, <i>false</i> otherwise. Examples of why this method fails include:</p> <ul style="list-style-type: none"> • No selection server for <i>type</i>. • Attempting to set the selection during link creation. |
| Exceptions | Only those exceptions raised by the previous selection server's AVDocSelectionGettingSelectionProc , and those raised by the new selection server's AVDocSelectionGettingSelectionProc . |

| | |
|------------------------|--|
| Notifications | AVDocDidPrint |
| Header File | <i>AVCalls.h</i> |
| Related Methods | ASAtomFromString PDDocCreateStructTreeRoot AVGrafSelectCreate AVPageViewDrawNow AVDocRegisterSelectionServer AVDocClearSelection AVDocDeleteSelection AVDocDoSaveAsWithParams AVDocEnumSelection AVDocCopySelection |

Example

```

/* Select text */
PDTextSelect TextSelection;
HiliteEntry Hilite;

PDPage CurrentPDPage =
    PDDocAcquirePage(CurrentPDDoc, PageNum);
Hilite.offset= (ASUnsl6) Offset;
Hilite.length= 0;
TextSelection =
    PDTextSelectCreatePageHilite(
        CurrentPDPage, &Hilite, 1);
AVDocSetSelection(CurrentAVDoc,
    ASAtomFromString("Text"), (void *)
    TextSelection, true);
AVDocShowSelection(CurrentAVDoc);
PDPageRelease(CurrentPDPage);

/* select 0th link annot on 0th page
(assumes that it s a link) */
#define k_Annot
    ASAtomFromString("Annotation")
CosObj tmp;
PDAnnot annot, *annotSel;
PDAction action;

DURING
    PDDocAcquirePage(pdDoc, 0); /* acquire
    0th page */
    annot = PDPageGetAnnot(pdPage, 0); /* get
    0th annot on pdPage */
    AnnotSel = ASmalloc(sizeof(PDAnnot));
    if(annotSel){
        tmp = PDActionGetCosObj(action); /*
        so we can copy its contents */
        *annotSel = tmp; /*copy the contents*/

        /* the selection server will ASfree
        the pointer when done with
        annotSel */
        AVDocSetSelection(avdoc, k_Annot,
            &annotSel, true); }
    PDPageRelease(pdPage);
HANDLER
    if(pdPage)
        PDPageRelease(pdPage);
END_HANDLER

```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocSetSplitterPosition

```
void AVDocSetSplitterPosition (AVDoc doc,  
    ASInt16 newPosition);
```

Description Sets the splitter position. The splitter is the vertical division between the bookmark/thumbnail pane and the document pane. The default splitter location is saved in the Acrobat viewer's preferences file, and can be read/set using [AVAppGetPreference](#) and [AVAppSetPreference](#).

Parameters

doc
The document whose splitter position is set.

newPosition
The new splitter position. This value specifies the width of the bookmark/thumbnail pane in pixels.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVDocGetSplitterPosition](#)
[AVAppGetPreference](#)

Example

```
if (AVDocGetSplitterPosition(doc) < 72)  
    AVDocSetSplitterPosition(doc, 72);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocSetViewDef

```
void AVDocSetViewDef (AVDoc doc, AVDocViewDef viewDef);
```

| | |
|------------------------|--|
| Description | Sets the document's state to match the information in <i>viewDef</i> . |
| Parameters | <p><i>doc</i></p> <p>The document whose state is updated.</p> <p><i>viewDef</i></p> <p>A pointer to the structure that stores the state.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocGetViewDef |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocSetViewMode

```
void AVDocSetViewMode (AVDoc doc, PDPageMode newMode);
```

| | |
|------------------------|--|
| Description | <p>Sets the current view mode.</p> <p>This method does nothing if the current view mode is full-screen, <i>PDFFullScreen</i>. In this case, call AVAppEndFullScreen first.</p> |
| Parameters | <p><i>doc</i></p> <p>The document whose view mode is set.</p> <p><i>newMode</i></p> <p>The view mode to set.</p> |
| Return Value | None. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppEndFullScreen AVDocGetViewMode |
| Example | <pre>if(AVDocGetViewMode(doc) == PDUseNone) AVDocSetViewMode(doc, PDUseBookmarks);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVDocShowSelection

```
void AVDocShowSelection (AVDoc doc);
```

| | |
|------------------------|--|
| Description | Displays the current selection by calling the selection server's AVDocSelectionShowSelectionProc callback. Does nothing if the document has no selection or the current selection's server has no AVDocSelectionShowSelectionProc callback. |
| Parameters | <i>doc</i> The document whose selection is shown. |
| Return Value | None |
| Exceptions | Only those raised by the selection server's AVDocSelectionShowSelectionProc callback. |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocSetSelection PDTextSelectCreatePageHilite PDTextSelectCreateWordHilite PDDocCreateStructTreeRoot AVGrafSelectCreate AVDocRegisterSelectionServer |
| Example | <pre>PDTextSelect TextSelection; HiliteEntry Hilite; PDPage CurrentPDPage = PDDocAcquirePage(CurrentPDDoc, PageNum); Hilite.offset= (ASUnsl6) Offset; Hilite.length= 0; TextSelection = PDTextSelectCreatePageHilite(CurrentPDPage, &Hilite, 1); AVDocSetSelection(CurrentAVDoc, ASAtomFromString("Text"), (void *) TextSelection, true); if(show) AVDocShowSelection(CurrentAVDoc); else AVDocClearSelection(CurrentAVDoc, true); PDPageRelease(CurrentPDPage);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVGrafSelect

AVGrafSelectCreate

```
AVGrafSelect AVGrafSelectCreate (AVPageView pageView,  
    const ASFixedRectP selRect);
```

| | |
|------------------------|--|
| Description | Creates a graphics selection. After creation, the selection can be set using AVDocSetSelection . |
| Parameters | <p><i>pageView</i></p> <p>The <i>AVPageView</i> in which a graphics selection is created.</p> <p><i>selRect</i></p> <p>Pointer to the ASFixedRect bounding the region from which a graphics selection is created, specified in user space coordinates.</p> |
| Return Value | The newly-created <i>AVGrafSelect</i> , or <i>NULL</i> if <i>selRect</i> is <i>NULL</i> or is an empty rectangle. |
| Exceptions | genErrBadParm genErrNoMemory |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVGrafSelectDestroy AVDocSetSelection |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVGrafSelectDestroy

```
void AVGrafSelectDestroy (AVGrafSelect avGraf);
```

Description Destroys a graphics selection. Use this method to destroy the selection only if you have *not* called [AVDocSetSelection](#) with it. If the selection has been set by a call to [AVDocSetSelection](#), it will automatically be destroyed by a call to [AVDocClearSelection](#) or the next call to [AVDocSetSelection](#).

Parameters *avGraf*
The graphics selection to destroy.

Return Value None

Exceptions [genErrBadParm](#)

Notifications None

Header File *AVCalls.h*

Related Methods [AVGrafSelectCreate](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVGrafSelectGetBoundingRect

```
void AVGrafSelectGetBoundingRect ( AVGrafSelect avGraf,
    ASFixedRect* boundRectP );
```

| | |
|------------------------|---|
| Description | Gets the specified graphics selection's bounding rectangle. |
| Parameters | <p><i>avGraf</i></p> <p>The graphics selection whose bounding rectangle is obtained.</p> <p><i>boundRectP</i></p> <p>Pointer to the graphics selection's bounding rectangle, specified in user space coordinates.</p> |
| Return Value | None |
| Exceptions | genErrBadParm |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVGrafSelectCreate |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenu

AVMenuAcquire

[AVMenu](#) AVMenuAcquire ([AVMenu](#) menu);

| | |
|------------------------|--|
| Description | Acquires the specified menu. Increments the menu's reference count. |
| Parameters | <i>menu</i> The menu to acquire. |
| Return Value | The menu acquired. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenubarAcquireMenuByIndex AVMenubarAcquireMenuByName AVMenubarAcquireMenuByPredicate AVMenuRelease AVMenuNew |

Example

```
AVMenu tempMenu;
ASInt32 i;

for(i=0; i< AVMenubarGetNumMenus(
    AVAppGetMenuBar()); i++)
{
    tempMenu = AVMenubarAcquireMenuByIndex(
        AVAppGetMenuBar(), i);
    if(AVMenuGetName(tempMenu) ==
        ASAtomFromString("Edit"))
        AVMenuRemove(tempMenu);
    break;
}
AVMenuRelease(tempMenu);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuAcquireMenuItemByIndex

[AVMenuItem](#) AVMenuAcquireMenuItemByIndex ([AVMenu](#) menu,
ASInt32 menuItemIndex);

| | |
|------------------------|--|
| Description | Acquires the menu item at the specified location in the specified menu. When you are done using the menu item, release it using AVMenuItemRelease . Menu item indices are generally not reliable—they change as plug-ins add, remove, or rearrange menu items, and may differ in different versions of the Acrobat viewer (if menu items are rearranged, removed, or added). Menu items should generally be acquired using AVMenubarAcquireMenuItemByName , which is generally reliable. |
| Parameters | <p><i>menu</i></p> <p>The menu in which a menu item is acquired.</p> <p><i>menuItemIndex</i></p> <p>The index of the menu item in <i>menu</i> to acquire. The first item in a menu has an index of zero. Even if the Acrobat viewer is displaying short menus, the index includes any intervening long-mode-only menu items (irrelevant for Acrobat 3.0 or later since there are no short menus).</p> |
| Return Value | The menu item whose index is specified. Returns <i>NULL</i> if <i>menu</i> is <i>NULL</i> , if the index is less than zero, or the index is greater than the number of menu items in the menu. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuAcquire AVMenuGetNumMenuItems AVMenubarAcquireMenuItemByPredicate AVMenubarAcquireMenuItemByName AVMenuItemAcquire AVMenuItemRelease |

Example

```
AVMenuItem tempItem;
ASInt32 i;

for(i=0;i< AVMenuGetNumMenuItems(
    editMenu); i++)
{
    tempItem =
        AVMenuAcquireMenuItemByIndex(
            editMenu, i);
    AVMenuItemGetTitle(tempItem, buf,
        sizeof(buf));
    if(!strcmp(buf, "Paste"))
        AVMenuItemSetTitle(tempItem, "Glue");
}
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuAddMenuItem

```
void AVMenuAddMenuItem (AVMenu menu, AVMenuItem menuItem,
    ASInt32 menuItemIndex);
```

Description Inserts a menu item in the specified location in a menu, and acquires the menu item. Does nothing if the menu is *NULL*, if the menu item is *NULL*, or the menu item is already in a menu.

Parameters

menu
The menu to which a menu item is added.

menuItem
The menu item to add.

menuItemIndex
The location in *menu* to add *menuItem*. The first item in a menu has an index of zero. Even if the Acrobat viewer is displaying short menus, the index includes any intervening long-mode-only menu items (irrelevant for Acrobat 3.0 or later since there are no short menus). Pass *APPEND_MENUITEM* (see *AVExpT.h*) to append the menu item to the end of the menu.

Return Value None

Exceptions [genErrNoMemory](#)

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuItemRemove](#)

Example

```
myButtonItem =
    AVMenubarAcquireMenuItemByName (menubar,
    "Hello");
helloIndex = AVMenuGetMenuItemIndex(menu,
    myButtonItem);
AVMenuAddMenuItem(menu, notesMenuItem,
    helloIndex+1);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuGetMenuItemIndex

```
ASInt32 AVMenuGetMenuItemIndex (AVMenu menu,
    AVMenuItem menuItem);
```

| | |
|------------------------|--|
| Description | <p>Gets the index of the specified menu item in the specified menu.</p> <p>Indices must be used with caution, because they change as plug-ins add, remove, or rearrange menu items.</p> |
| Parameters | <p><i>menu</i></p> <p>The menu in which <i>menuItem</i> is located.</p> <p><i>menuItem</i></p> <p>The menu item whose index is obtained.</p> |
| Return Value | <p>The index of <i>menuItem</i> in <i>menu</i>. The first item in a menu has an index of zero. Even if the Acrobat viewer is displaying short menus, the index includes any intervening long-mode-only menu items (irrelevant for Acrobat 3.0 or later since there are no short menus).</p> <p>Returns <i>BAD_MENUITEM_INDEX</i> (see <i>AVExpT.h</i>) if <i>menuItem</i> is not in <i>menu</i>. Also returns <i>BAD_MENUITEM_INDEX</i> if <i>menu</i> is <i>NULL</i> or <i>menuItem</i> is <i>NULL</i>.</p> |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuGetNumMenuItems |
| Example | <pre>myButtonItem = AVMenubarAcquireMenuItemByName (menubar , "Hello"); helloIndex = AVMenuGetMenuItemIndex(menu, myButtonItem); AVMenuAddMenuItem(menu, notesMenuItem, helloIndex+1);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuGetName

[ASAtom](#) AVMenuGetName ([AVMenu](#) menu);

| | |
|------------------------|---|
| Description | Gets the <i>ASAtom</i> for the menu's language-independent name. |
| Parameters | <p><i>menu</i></p> <p>The menu whose language-independent name is obtained.</p> |
| Return Value | The menu's name. The <i>ASAtom</i> can be converted to a string using ASAtomGetString . Returns <i>NULL</i> if <i>menu</i> is <i>NULL</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuAcquire AVMenuGetTitle AVMenuNew AVMenuItemGetName |
| Example | <pre>AVMenu tempMenu; ASInt32 i; for(i=0;i< AVMenubarGetNumMenus(AVAppGetMenuBar());i++){ tempMenu = AVMenubarAcquireMenuByIndex(AVAppGetMenuBar(), i); if(AVMenuGetName(tempMenu) == ASAtomFromString("Edit")) { AVMenuRemove(tempMenu); break; } } AVMenuRelease(tempMenu);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuGetNumMenuItems

```
ASInt32 AVMenuGetNumMenuItems (AVMenu menu);
```

| | |
|------------------------|---|
| Description | Gets the number of menu items in a menu, including those that are visible only in long-menus mode. |
| Parameters | <p><i>menu</i></p> <p>The menu for which the number of menu items is obtained.</p> |
| Return Value | The number of menu items in the specified menu. Returns 0 if <i>menu</i> is <i>NULL</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuGetMenuItemIndex AVMenuAcquireMenuItemByIndex |
| Example | <pre>if (AVMenuGetNumMenuItems(myMenu)) { /* nonzero means process it */ }</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuGetParentMenubar

[AVMenubar](#) AVMenuGetParentMenubar ([AVMenu](#) menu) ;

| | |
|------------------------|---|
| Description | Gets the parent menu bar for the specified menu. |
| Parameters | <p><i>menu</i></p> <p>The menu whose parent menu bar is obtained.</p> |
| Return Value | The menu bar to which the menu is attached. Returns <i>NULL</i> if the menu has not yet been added to the menu bar or if the menu is a submenu. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuGetParentMenuItem AVMenubarGetMenuIndex AVMenuItemGetParentMenu |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuGetParentMenuItem

[AVMenuItem](#) AVMenuGetParentMenuItem ([AVMenu](#) menu) ;

| | |
|------------------------|--|
| Description | Gets the parent menu item for the specified menu. |
| Parameters | <p><i>menu</i></p> <p>The menu whose parent menu item is obtained.</p> |
| Return Value | The menu item for which the specified menu is a submenu. Returns <i>NULL</i> if the specified menu is not a submenu. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuGetParentMenubar AVMenubarGetMenuIndex AVMenuItemGetParentMenu |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuGetTitle

```
ASInt32 AVMenuGetTitle (AVMenu menu, char* buffer,
    ASInt32 bufferSize);
```

Description Gets the menu's title as it appears in the user interface. The length of the title remains 0 if menu is *NULL*.

Parameters

menu

The menu whose title is obtained.

buffer

(Filled by the method) The buffer into which the title is copied. If *buffer* is *NULL*, it is not filled but the length of the title is still returned.

bufferSize

The maximum number of characters the buffer can hold.

Return Value If *menu* is nonzero, returns the length of the title. If *menu* is *NULL* and *buffer* is not, *buffer* is zeroed. Returns zero if *buffer* is *NULL*.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuGetName](#)
[AVMenuItemGetTitle](#)

Example

```
AVMenu tempMenu;
ASInt32 i;

for(i=0;i< AVMenubarGetNumMenus(
    AVAppGetMenuBar());i++){
    tempMenu = AVMenubarAcquireMenuByIndex(
        AVAppGetMenuBar(), i);
    AVMenuGetTitle(tempMenu, buf,
        sizeof(buf));
    if(!strcmp(buf, "Edit")){
        AVMenuSetTitle(tempMenu, "Change");
        AVMenuRelease(tempMenu);
        break;
    }
    AVMenuRelease(tempMenu);
}
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuIsHiddenOnMenubar

```
ASBool AVMenuIsHiddenOnMenubar (AVMenu menu);
```

| | |
|------------------------|---|
| Description | Tests whether a menu is hidden on the menubar. |
| Parameters | <i>menu</i> The menu to test. |
| Return Value | <i>true</i> if menu is hidden, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenubarAddHiddenMenu |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVMenuNew

[AVMenu](#) AVMenuNew (const char* title, const char* name,
[ASExtension](#) owner);

| | |
|------------------------|---|
| Description | Creates and acquires a new menu with the given title and language-independent name. The menu can be added to the menu bar using AVMenubarAddMenu . When you are done using the menu, release it using AVMenuRelease . |
| Parameters | <p><i>title</i></p> <p>The string that appears in the user interface. In Windows, an ampersand (&) character in the string results in underlining the character after it on the menu.</p> <p><i>name</i></p> <p>Language-independent name of the menu to create. This is the value returned by AVMenuGetName. <i>name</i> must <i>not</i> contain any spaces. Plug-in developers should prefix the names of menus they add with the name of their plug-in and a colon, to avoid collisions in the menu name space. For example, a plug-in named <i>myPlug</i> might add menus named <i>myPlug:DrawTools</i> and <i>myPlug:Checkout</i>.</p> <p><i>owner</i></p> <p>The gExtensionID extension registering the menu.</p> |
| Return Value | The newly-created menu. |
| Exceptions | genErrNoMemory |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuRelease AVMenubarAddMenu AVMenuGetName |
| Example | <pre>AVMenu notesMenu; notesMenu = AVMenuNew("Notes", "NewNotes", gExtensionID);</pre> |

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuRelease

```
void AVMenuRelease (AVMenu menu);
```

Description Releases the specified menu. Decrements the reference count and automatically destroys the menu when its reference count is zero.

Parameters *menu*
The menu to release.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuAcquire](#)
[AVMenuNew](#)

Example

```
AVMenu tempMenu;
ASInt32 i;

for(i=0; i< AVMenubarGetNumMenus(
    AVAppGetMenuBar()); i++)
{
    tempMenu = AVMenubarAcquireMenuByIndex(
        AVAppGetMenuBar(), i);
    if(AVMenuGetName(tempMenu) ==
        ASAtomFromString("Edit"))
        AVMenuRemove(tempMenu);
    break;
}
AVMenuRelease(tempMenu);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

AVMenuRemove

```
void AVMenuRemove (AVMenu menu);
```

Description Removes a menu from the menu bar and releases it. If the menu is a submenu, this method does nothing.

Parameters *menu*
The menu to remove.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuItemRemove](#)
[AVMenuRelease](#)

Example

```
AVMenu tempMenu;
ASInt32 i;

for(i=0;i< AVMenubarGetNumMenus(
    AVAppGetMenuBar());i++){
    tempMenu =
        AVMenubarAcquireMenuByIndex(
            AVAppGetMenuBar(), i);
    if(AVMenuGetName(tempMenu) ==
        ASAtomFromString("Edit")) {
        AVMenuRemove(tempMenu);
        break;
    }
    AVMenuRelease(tempMenu);
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

AVMenubar

AVMenubarAcquireMenuByIndex

```
AVMenu AVMenubarAcquireMenuByIndex (AVMenubar menubar ,
ASInt32 menuIndex);
```

| | |
|------------------------|--|
| Description | Acquires the menu with the specified index. Menu indices are generally not reliable—they change as plug-ins add, remove, or rearrange menus, and may differ in different versions of the Acrobat viewer (if menus are rearranged, removed, or added). Menus should generally be acquired using AVMenubarAcquireMenuByName , which is generally reliable. |
| Parameters | <p><i>menubar</i></p> <p>The menu bar in which the menu is located.</p> <p><i>menuIndex</i></p> <p>The index (in <i>menubar</i>) of the menu to acquire.</p> |
| Return Value | The menu with the specified index. Returns <i>NULL</i> if no such menu or if <i>menubar</i> is <i>NULL</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenubarAcquireMenuByName AVMenubarAcquireMenuByPredicate AVAppGetMenubar AVMenuRelease |
| Example | <pre>AVMenu fileMenu; fileMenu = AVMenubarAcquireMenuByIndex(AVAppGetMenuBar(), 1); if (AVMenubarGetMenuIndex(fileMenu) != 1) AAlertNote("Odd."); AVMenuRelease(fileMenu);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenubarAcquireMenuByName

[AVMenu](#) AVMenubarAcquireMenuByName ([AVMenubar](#) menubar,
const char* name);

Description Acquires the menu or submenu that has the specified language-independent menu name (case-sensitive). When you are done using the menu, release it using [AVMenuRelease](#). Acquiring a menu by name is generally reliable, because names (unlike indices) do not change as menus are added or rearranged.

Parameters

menubar
The menu bar in which the menu is located.

name
The language-independent name of the menu to acquire. See [Menu Names](#) for a list of the names of the built-in menus in Acrobat Exchange.

Return Value The menu with the specified name.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppGetMenubar](#)
[AVMenuRelease](#)
[AVMenubarAcquireMenuByPredicate](#)
[AVMenubarAcquireMenuByIndex](#)

Example

```
menu = AVMenubarAcquireMenuByName(menubar,  
    "Tools");  
if (menu){  
    myButtonMenuItem = AVMenuItemNew( );  
  
    AVMenuRelease(menu);  
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenubarAcquireMenuByPredicate

[AVMenu](#) AVMenubarAcquireMenuByPredicate ([AVMenubar](#) menubar, [AVMenuPredicate](#) predicate, void* clientData);

Description Acquires a menu using a user-supplied selection routine. This method can also be used to enumerate all menus. When you are done using the menu that is acquired, release it using [AVMenuRelease](#).

Parameters *menubar*

The menu bar containing the menu to acquire.

predicate

User-supplied [AVMenuPredicate](#) function that determines which menu is acquired. Menus are searched depth-first. The first menu for which *predicate* returns *true* is acquired. If *predicate* always returns *false*, all menus will be enumerated.

clientData

Pointer to user-supplied data to pass to *predicate* each time it is called.

Return Value The first menu for which *predicate* returned *true*. Returns *NULL* if *predicate* never returned *true* or if *menubar* is *NULL*.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuRelease](#)
[AVMenubarAcquireMenuByName](#)
[AVMenubarAcquireMenuByIndex](#)
[AVAppGetMenubar](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenubarAcquireMenuItemByName

```
AVMenuItem AVMenubarAcquireMenuItemByName (AVMenubar menubar,  
const char* name);
```

Description Acquires the menu item with the specified language-independent menu item name (case-sensitive). This method automatically searches all menus and submenus. When you are done using the menu item, release it using [AVMenuItemRelease](#). Acquiring a menu item by name is generally reliable, because names (unlike indices) do not change as menus items are added or rearranged.

Parameters *menubar*
The menu bar in which the menu item is located.

name
The language-independent name of the menu item to acquire. See the tables on pages [2145](#) to [2159](#) for the language-independent names of the menu items built into Reader and Exchange. The language-independent names of menu items added by Adobe plug-ins are described in the technical notes for those plug-ins.

Return Value The menu item with the specified name. Returns *NULL* if no such menu item exists, if *menubar* is *NULL*, or if *name* is *NULL*.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppGetMenubar](#)
[AVMenuItemRelease](#)
[AVMenubarAcquireMenuItemByPredicate](#)
[AVMenuAcquireMenuItemByIndex](#)

Example

```
myButtonMenuItem =  
    AVMenubarAcquireMenuItemByName(menubar,  
    "Hello");  
helloIndex = AVMenuGetMenuItemIndex(menu,  
    myButtonMenuItem);  
AVMenuAddMenuItem(menu, notesMenuItem,  
    helloIndex+1);
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenubarAcquireMenuItemByPredicate

```
AVMenuItem AVMenubarAcquireMenuItemByPredicate
(AVMenubar menubar, AVMenuItemPredicate predicate,
void* clientData);
```

Description Acquires a menu item using a user-supplied selection routine. This method may also be used to enumerate all menu items. When you are done using the menu item that is acquired, release it using [AVMenuItemRelease](#).

Parameters

menubar
The menu bar containing the menu to acquire.

predicate
User-supplied function that determines which menu item is acquired. Menus items are searched depth-first. The first menu item for which *predicate* returns *true* is acquired. If *predicate* always returns *false*, all menu items will be enumerated.

clientData
Pointer to user-supplied data to pass to *predicate* each time it is called.

Return Value The first menu item for which *predicate* returned *true*. Returns *NULL* if predicate never returns *true* or if *menubar* is *NULL*.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuItemRelease](#)
[AVMenubarAcquireMenuItemByName](#)
[AVMenuAcquireMenuItemByIndex](#)
[AVAppGetMenubar](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenubarAddHiddenMenu

```
void AVMenubarAddHiddenMenu ( AVMenubar menubar, AVMenu menu);
```

Description Inserts a hidden menu into the menu bar. Does nothing if *menubar* is *NULL* or *menu* is *NULL*.

Parameters

menubar
The menu bar into which *menu* is added.

menu
The menu to add to *menubar*.

Return Value None

Exceptions [genErrNoMemory](#)

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuIsHiddenOnMenubar](#)
[AVMenubarAddMenu](#)
[AVMenuNew](#)
[AVMenuRemove](#)

Example

```
AVMenu hiddenNotesMenu;

hiddenNotesMenu = AVMenuNew("Notes",
    "NewNotes", gExtensionID);
AVMenubarAddHiddenMenu(AVAppGetMenuBar(),
    hiddenNotesMenu);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVMenubarAddMenu

```
void AVMenubarAddMenu ( AVMenubar menubar, AVMenu menu,
                        ASInt32 menuIndex);
```

| | |
|------------------------|---|
| Description | Inserts a menu into the menu bar. Does nothing if <i>menubar</i> is <i>NULL</i> or <i>menu</i> is <i>NULL</i> . |
| Parameters | <p><i>menubar</i></p> <p>The menu bar into which <i>menu</i> is added.</p> <p><i>menu</i></p> <p>The menu to add to <i>menubar</i>.</p> <p><i>menuIndex</i></p> <p>The position at which the menu is added. The leftmost menu in a menu bar has an index of zero. Passing a value of <i>APPEND_MENU</i> (see <i>AVExpT.h</i>) adds the menu to the end of the menu bar.</p> |
| Return Value | None |
| Exceptions | genErrNoMemory |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuNew AVMenuRemove AVMenubarAddHiddenMenu |
| Example | <pre>AVMenu notesMenu; notesMenu = AVMenuNew("Notes", "NewNotes", gExtensionID); AVMenubarAddMenu(AVAppGetMenuBar(), notesMenu, 3);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenubarGetMenuIndex

```
ASInt32 AVMenubarGetMenuIndex (AVMenubar menubar ,  
    AVMenu menu) ;
```

| | |
|------------------------|---|
| Description | Gets the index of the specified menu in the menu bar. |
| Parameters | <i>menubar</i> The menu bar in which <i>menu</i> is located. <i>menu</i> The menu whose index is obtained. |
| Return Value | The specified menu's index. Returns <i>BAD_MENU_INDEX</i> if the menu is not in the menu bar, is a submenu, if <i>menubar</i> is <i>NULL</i> , or if <i>menu</i> is <i>NULL</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppGetMenubar |
| Example | <pre>AVMenu fileMenu; fileMenu = AVMenubarAcquireMenuByIndex(AVAppGetMenuBar(), 1); if(AVMenubarGetMenuIndex(fileMenu) != 1) AVAlertNote("Odd.");</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenubarGetNumMenus

```
ASInt32 AVMenubarGetNumMenus ( AVMenubar menubar );
```

| | |
|------------------------|---|
| Description | Gets the number of menus in <i>menubar</i> . |
| Parameters | <p><i>menubar</i></p> <p>The menu bar for which the number of menus is obtained.</p> |
| Return Value | The number of menus in the menu bar, not including submenus. Returns 0 if <i>menubar</i> is <i>NULL</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppGetMenubar |
| Example | <pre>AVMenubar bar = AVAppGetMenubar(); ASInt32 numMenus = AVMenubarGetNumMenus(bar);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenubarHide

```
void AVMenubarHide ( AVMenubar menubar );
```

Description Hides the menu bar.

Parameters *menubar*
The menu bar to hide.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenubarShow](#)
[AVAppGetMenubar](#)

Example `AVMenubarHide (AVAppGetMenubar ());`

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenubarShow

```
void AVMenubarShow ( AVMenubar menubar );
```

Description Shows the menu bar.

Parameters *menubar*
The menu bar to show.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenubarHide](#)
[AVAppGetMenubar](#)

Example `AVMenubarShow(AVAppGetMenubar());`

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItem

AVMenuItemAcquire

[AVMenuItem](#) AVMenuItemAcquire ([AVMenuItem](#) menuItem);

Description Acquires a menu item. Increments the menu item's reference count.

Parameters *menuItem*
The menu item to acquire.

Return Value The menu item acquired.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuItemRelease](#)
[AVMenuBarAcquireMenuItemByPredicate](#)
[AVMenuBarAcquireMenuItemByName](#)
[AVMenuAcquireMenuItemByIndex](#)

Example

```
AVMenuItem tempItem;
ASInt32 i;

for(i=0;i< AVMenuGetNumMenuItems(
    editMenu);i++){
    tempItem = AVMenuAcquireMenuItemByIndex(
        editMenu, i);
    if(ASAtomFromString("Paste") ==
        AVMenuItemGetName(tempItem)){
        AVMenuItemRemove(tempItem);
        break;
    }
    AVMenuItemRelease(tempItem);
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemAcquireSubmenu

[AVMenu](#) AVMenuItemAcquireSubmenu ([AVMenuItem](#) menuItem);

Description Acquires the submenu attached to the specified menu item, if there is one. When you are done with the submenu, release it using [AVMenuRelease](#).

Parameters *menuItem*
The menu items whose submenu is obtained.

Return Value The specified menu item's submenu. Returns *NULL* if *menuItem* is *NULL* or if *menuItem* does not have a submenu.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuRelease](#)

Example

```
AVMenuItem tempItem;
AVMenu prefsMenu;
ASInt32 i;
buf[20];

for(i=0;i< AVMenuGetNumMenuItems(
    editMenu);i++){
    tempItem =
        AVMenuAcquireMenuItemByIndex(
            editMenu, i);
    AVMenuItemGetTitle(tempItem, buf,
        sizeof(buf));
    if(!strcmp(buf, "Preferences"))
        break;
    AVMenuItemRelease(tempItem);
}
prefsMenu = AVMenuItemAcquireSubmenu(
    tempItem);
/* now add items to prefsMenu */
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemExecute

```
void AVMenuItemExecute (AVMenuItem menuItem);
```

Description Executes a menu item's [AVExecuteProc](#). Does nothing if *menuItem* is *NULL*, if *menuItem* has no [AVExecuteProc](#), or if *menuItem* is not enabled.

You cannot execute a menu item that has a submenu (for example, the "Pages" menu item in the Edit menu).

Parameters *menuItem*

The menu item to execute.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuItemSetExecuteProc](#)
[AVMenuItemIsEnabled](#)

Example

```
AVMenuItem oldItem;

if(oldItem)
    AVMenuItemExecute(oldItem);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemGetLongOnly

```
ASBool AVMenuItemGetLongOnly (AVMenuItem menuItem);
```

Description Gets the flag indicating whether a menu item is visible only in long-menus mode.

Parameters *menuItem*
The menu item whose flag is obtained.

Return Value *true* if the menu item is visible only in long-menus mode. *false* if the menu item is visible in both long and short menus, or if *menuItem* is *NULL*.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuAcquireMenuItemByIndex](#)
[AVMenubarAcquireMenuItemByName](#)
[AVMenubarAcquireMenuItemByPredicate](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemGetName

[ASAtom](#) AVMenuItemGetName ([AVMenuItem](#) menuItem);

Description Gets the atom for the language-independent name of the menu item.

Parameters *menuItem*
The menu item whose language-independent name is obtained.

Return Value The *ASAtom* corresponding to the name of the specified menu item, or *ASAtomNull* if *menuItem* is *NULL*. The *ASAtom* can be converted to a string using [ASAtomGetString](#).

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuItemGetTitle](#)

Example

```
AVMenuItem tempItem;
ASInt32 i;

for(i=0;i< AVMenuGetNumMenuItems(
    editMenu);i++){
    tempItem =
        AVMenuAcquireMenuItemByIndex(
            editMenu, i);
    if(ASAtomFromString("Paste") ==
        AVMenuItemGetName(tempItem)){
        AVMenuItemSetTitle(tempItem, "Glue");
        AVMenuItemRelease(tempItem);
        break;
    }
    AVMenuItemRelease(tempItem);
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemGetParentMenu

[AVMenu](#) AVMenuItemGetParentMenu ([AVMenuItem](#) menuItem) ;

| | |
|------------------------|--|
| Description | Gets the menu in which the specified menu item appears. |
| Parameters | <p><i>menuItem</i></p> <p>The menu item whose parent menu is obtained.</p> |
| Return Value | The menu in which the specified menu item appears. Returns <i>NULL</i> if this menu item is not in a menu. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuGetParentMenuItem AVMenubarGetMenuIndex AVMenuItemAcquireSubmenu |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemGetShortcut

```
ASBool AVMenuItemGetShortcut (AVMenuItem menuItem, char* key,
                              ASInt16* flags);
```

Description Gets the shortcut key for the specified menu item.

Parameters

menuItem
The menu item whose shortcut is obtained.

key
(Filled by the method) The key that is a shortcut for the menu item, an ASCII character. The value *NO_SHORTCUT* (see *AVExpT.h*) indicates that the menu item has no shortcut.

flags
(Filled by the method) Modifier keys, if any, used as part of the shortcut. Must be an OR of the [Modifier Keys](#) values, except that *AV_COMMAND* will never be present.

Return Value *true* if *menuItem* is not *NULL* and *menuItem* has a shortcut, *false* otherwise.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuGetMenuItemIndex](#)
[AVMenuItemNew](#)

Example

```
AVMenuItem item;
ASInt16 flags;
char* key;
if (AVMenuItemGetShortcut(item, key,
    &flags))
    AAlertNote("shortcut exists");
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemGetTitle

```
ASInt32 AVMenuItemGetTitle (AVMenuItem menuItem, char* buffer,
    ASInt32 bufferSize);
```

| | |
|------------------------|--|
| Description | Gets a menu item's title, which is the string that appears in the user interface. |
| Parameters | <p><i>menuItem</i></p> <p>The menu item whose title is obtained.</p> <p><i>buffer</i></p> <p>(Filled by the method) A buffer to hold the <i>NULL</i>-terminated name. If <i>title</i> is <i>NULL</i>, returns the length of the menu item's name, but does not fill the buffer.</p> <p><i>bufferSize</i></p> <p>The maximum number of characters that can be placed into <i>title</i>. If the name is longer than this, only the first <i>bufferSize</i> – 1 characters are placed into <i>buffer</i>, and the buffer is <i>NULL</i>-terminated.</p> |
| Return Value | The length of the title. Returns 0 if <i>menuItem</i> is <i>NULL</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuItemSetTitle AVMenuItemGetName |
| Example | <pre>AVMenuItem tempItem; ASInt32 i; buf[20]; for(i=0;i< AVMenuGetNumMenuItems(editMenu);i++){ tempItem = AVMenuAcquireMenuItemByIndex(editMenu, i); AVMenuItemGetTitle(tempItem, buf, sizeof(buf)); if(!strcmp(buf, "Paste")) AVMenuItemSetTitle(tempItem, "Glue"); AVMenuItemRelease(tempItem); }</pre> |

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemIsEnabled

```
ASBool AVMenuItemIsEnabled ( AVMenuItem menuItem );
```

| | |
|------------------------|---|
| Description | Tests whether or not the specified menu item is enabled. |
| Parameters | <p><i>menuItem</i></p> <p>The menu item whose enabled flag is obtained.</p> |
| Return Value | <p><i>true</i> if <i>menuItem</i> is enabled, if <i>menuItem</i> is <i>NULL</i>, or if <i>menuItem</i> has no AVComputeEnabledProc.</p> <p><i>false</i> if the menu item is disabled or its AVComputeEnabledProc raises an exception.</p> |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuItemSetComputeEnabledProc |
| Example | <pre>if (AVMenuItemIsEnabled(item) AVMenuItemExecute(item);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemIsMarked

```
ASBool AVMenuItemIsMarked (AVMenuItem menuItem);
```

| | |
|------------------------|--|
| Description | Tests whether or not <i>menuItem</i> is marked (for example, appears with a checkmark). |
| Parameters | <p><i>menuItem</i></p> <p>The menu item whose marked state is obtained.</p> |
| Return Value | <i>true</i> if <i>menuItem</i> is marked. <i>false</i> if <i>menuItem</i> is <i>NULL</i> , if the menu item does not have an AVComputeMarkedProc , or if it raises an exception. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuItemSetComputeMarkedProc |
| Example | <pre>if (AVMenuItemIsMarked(item)) AVMenuItemExecute(item);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemNew

```
AVMenuItem AVMenuItemNew (const char* title, const char* name,  
    AVMenu submenu, ASBool longMenusOnly, char shortcut,  
    ASInt16 flags, AVIcon icon, ASExtension owner);
```

Description

Creates and acquires a new *AVMenuItem*. The menu item can be added to a menu using [AVMenuAddMenuItem](#).

Release the *AVMenuItem* using [AVMenuItemRelease](#) after it has been added to a menu.

Parameters

title

The string shown in the user interface for this menu item. Use a hyphen to create a separator menu item. This value is also returned by [AVMenuItemGetTitle](#). In Windows, an ampersand (&) character in the string results in underlining the character after it on the menu item.

name

The language-independent name of the menu item to create. This is the value returned by [AVMenuItemGetName](#). *name* must *not* contain any spaces. Plug-in developers should prefix the names of menu items they add with the name of their plug-in and a colon, to avoid collisions in the menu item name space. For example, a plug-in named *myPlug* might add menu items named *myPlug:Scan* and *myPlug:Find*.

submenu

Submenu (if any) for which this menu item is the parent. Pass *NULL* if this menu item does not have a submenu.

longMenusOnly

(Ignored in Acrobat 3.0 or later) If *true*, the menu item is visible only when the user selects "Full Menus." If *false*, the menu item is visible for both "Full Menus" and "Short Menus" modes.

shortcut

The key to use as a shortcut for the menu item, an ASCII character. Use *NO_SHORTCUT* (see *AVExpT.h*) if the menu item has no shortcut.

The Acrobat viewer does not check for conflicts between shortcuts. The consequences of multiple menu items having the same shortcut is undefined.

In Windows, the shortcut is not displayed for any menu item that also has an *icon*, although the shortcut will work.

flags

Modifier keys, if any, used as part of the shortcut. Must be an OR of the [Modifier Keys](#) values, except that *AV_COMMAND* cannot be specified.

icon

The icon to show in the menu item, or *NULL* if no icon is shown.

In Mac OS, *icon* is a handle to a standard *SICN* resource.

In Windows, *icon* is a 24×24 sample monochrome *HBITMAP*.

owner

The [gExtensionID](#) extension registering the menu item.

| | |
|------------------------|--|
| Return Value | The newly-created menu item. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuAddMenuItem AVMenuItemRelease |

Example

```
ASCallback doHelloAlertCallback;
AVMenuItem myButtonMenuItem;

doHelloAlertCallback =
    ASCallbackCreateProto(AVExecuteProc,
        &doHelloAlert);
/* shortcut is CTRL-SHIFT-H */
myButtonMenuItem = AVMenuItemNew(
    "&Hello World", "Hello", NULL, false,
    H , AV_CONTROL || AV_SHIFT, NULL,
    gExtensionID);
AVMenuItemSetExecuteProc(myButtonMenuItem,
    doHelloAlertCallback, NULL);
```

Availability

Plug-ins: Available if *PL_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemRelease

```
void AVMenuItemRelease (AVMenuItem menuItem);
```

Description Releases a menu item. Decrements the reference count and destroys the menu item if the count is zero.

Parameters *menuItem*
The menu item to release.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuItemAcquire](#)
[AVMenuBarAcquireMenuItemByPredicate](#)
[AVMenuBarAcquireMenuItemByName](#)
[AVMenuAcquireMenuItemByIndex](#)

Example

```
AVMenuItem tempItem;
ASInt32 i;

for(i=0;i< AVMenuGetNumMenuItems(editMenu)
;i++){
tempItem =
    AVMenuAcquireMenuItemByIndex(
        editMenu, i);
if(ASAtomFromString("Paste") ==
    AVMenuItemGetName(tempItem)){
    AVMenuItemRemove(tempItem);
    break;
}
AVMenuItemRelease(tempItem);
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemRemove

```
void AVMenuItemRemove (AVMenuItem menuItem);
```

Description Removes a menu item from the menu hierarchy and releases it. Does nothing if *menuItem* is *NULL*.
Keyboard accelerators for the Acrobat viewer's built-in menu items will always work, even if the menu item is removed.

Parameters *menuItem*
The menu item to remove.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuItemAcquire](#)
[AVMenuItemRelease](#)

Example

```
ASInt32 i;  
  
for(i=0;i< AVMenuGetNumMenuItems(  
    editMenu);i++){  
    tempItem = AVMenuAcquireMenuItemByIndex(  
        editMenu, i);  
    if(ASAtomFromString("Paste") ==  
        AVMenuItemGetName(tempItem)){  
        AVMenuItemRemove(tempItem);  
        break;  
    }  
    AVMenuItemRelease(tempItem);  
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemSetComputeEnabledProc

```
void AVMenuItemSetComputeEnabledProc (AVMenuItem menuItem,
    AVComputeEnabledProc proc, void* data);
```

Description Sets the user-supplied procedure to call to determine whether or not the menu item is enabled. Does nothing if *menuItem* is *NULL*.

Parameters

menuItem

The menu item whose [AVComputeEnabledProc](#) is set.

proc

User-supplied callback to call whenever the Acrobat viewer needs to know whether or not *menuItem* should be enabled.

data

Pointer to user-supplied data to pass to *proc* each time it is called.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuItemIsEnabled](#)

Example

```
ASCallback doHelloAlertCallback;
AVMenuItem myButtonMenuItem;
```

```
doHelloAlertCallback =
    ASCallbackCreateProto(AVExecuteProc,
        &doHelloAlert);
myButtonMenuItem = AVMenuItemNew(
    "Hello World", "Hello", NULL, false,
    NO_SHORTCUT, 0, NULL, gExtensionID);
AVMenuItemSetComputeEnabledProc(
    myButtonMenuItem,
    DocExistEnableCallback, NULL);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemSetComputeMarkedProc

```
void AVMenuItemSetComputeMarkedProc (AVMenuItem menuItem,
    AVComputeMarkedProc proc, void* data);
```

| | |
|------------------------|--|
| Description | Sets the user-supplied procedure that determines whether or not the menu item appears with a checkmark. Does nothing if <i>menuItem</i> is <i>NULL</i> . If the menu item has no AVExecuteProc (see AVMenuItemSetExecuteProc), its AVComputeMarkedProc is never called. To avoid this, add an AVExecuteProc that does nothing, and if you wish the menu item to gray out, also add an AVComputeEnabledProc that always returns <i>false</i> . |
| Parameters | <p><i>menuItem</i></p> <p>The menu item whose AVComputeMarkedProc is being set.</p> <p><i>proc</i></p> <p>User-supplied callback to call whenever the Acrobat viewer needs to know whether or not the <i>menuItem</i> should be marked.</p> <p><i>data</i></p> <p>Pointer to user-supplied data to pass to <i>proc</i> each time it is called.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuItemIsMarked AVMenuItemSetExecuteProc AVMenuItemSetComputeEnabledProc |

Example

```
ASCallback markedProcCallback;
AVMenuItem myButtonMenuItem;

markedProcCallback =
    ASCallbackCreateProto(AVExecuteProc,
        &markedProc);
myButtonMenuItem = AVMenuItemNew(
    "Hello World", "Hello", NULL, false,
    NO_SHORTCUT, 0, NULL, gExtensionID);
AVMenuItemSetComputeMarkedProc(
    myButtonMenuItem, markedProcCallback,
    NULL);
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemSetExecuteProc

```
void AVMenuItemSetExecuteProc (AVMenuItem menuItem,
    AVExecuteProc proc, void* data);
```

| | |
|------------------------|---|
| Description | Sets the user-supplied procedure to execute whenever the menu item is chosen. Does nothing if <i>menuItem</i> is <i>NULL</i> . Plug-ins must not set the execute procedure of the Acrobat viewer's built-in menu items. |
| Parameters | <p><i>menuItem</i></p> <p>The menu item whose execute procedure is set.</p> <p><i>proc</i></p> <p>User-supplied callback to call whenever <i>menuItem</i> is selected.</p> <p><i>data</i></p> <p>Pointer to user-supplied data to pass to <i>proc</i> each time it is called.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVMenuItemExecute AVMenuItemSetComputeMarkedProc AVMenuItemSetComputeEnabledProc |
| Example | <pre>ASCallback markedProcCallback; AVMenuItem myButtonMenuItem; doHelloAlertCallback = ASCallbackCreateProto(AVExecuteProc, &doHelloAlert); myButtonMenuItem = AVMenuItemNew("Hello World", "Hello", NULL, false, NO_SHORTCUT, 0, NULL, gExtensionID); AVMenuItemSetExecuteProc(myButtonMenuItem, doHelloAlertCallback, NULL);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVMenuItemSetTitle

```
void AVMenuItemSetTitle (AVMenuItem menuItem,
    const char* title);
```

Description Sets a menu item's title, which is the string that appears in the user interface. Use this method to manage menu items whose titles change (such as "show/hide fooWindow"), instead of inserting and removing menu items on the fly.

Parameters

menuItem

The menu item whose title is set.

title

The new menu title. It must be a *NULL*-terminated string.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVMenuItemGetTitle](#)
[AVMenuItemGetName](#)

Example

```
AVMenuItem tempItem;
ASInt32 i;
char buf[20];

for(i=0;i< AVMenuGetNumMenuItems(
    editMenu);i++){
    tempItem =
        AVMenuAcquireMenuItemByIndex(
            editMenu, i);
    AVMenuItemSetTitle(tempItem, buf,
        sizeof(buf));
    if(!strcmp(buf, "Paste"))
        AVMenuItemSetTitle(tempItem, "Glue");
    AVMenuItemRelease(tempItem);
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageView

AVPageViewAcquireMachinePort

```
void* AVPageViewAcquireMachinePort (AVPageView pageView);
```

Description Acquires the platform-specific object needed to draw into the Acrobat viewer's document window using a platform's native graphics calls. When done, release it using [AVPageViewReleaseMachinePort](#).

Parameters *pageView*
The *AVPageView* whose platform-dependent port is acquired.

Return Value A platform-dependent value.

- In Mac OS, it is a *GrafPtr*.
- In Windows, it is a [WinPort](#).
- In UNIX, it is a *Widget*.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewAcquireMachinePort](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewAppearanceGetAVMatrix

```
void AVPageViewAppearanceGetAVMatrix (AVPageView PageView,
    ASUns32 flags, CosObj appear, ASFixedRect* ar,
    ASFixedMatrix* mr);
```

| | |
|------------------------|---|
| Description | Calculates a matrix for use with AVPageViewDrawCosObj or AVPageViewDrawCosObjEx that leaves the appearance unrotated and unzoomed as specified by <i>flags</i> . This is typically used in conjunction with drawing an annotation appearance represented by <i>appear</i> . |
| Parameters | <p><i>PageView</i></p> <p>The page view for which the matrix is calculated.</p> <p><i>flags</i></p> <p>Annotation flags obtained by <i>PDAnnotGetFlags</i>.</p> <p><i>appear</i></p> <p>Appearance of the object, which is a Form <i>XObject</i>.</p> <p><i>ar</i></p> <p>Bounding rectangle for <i>appear</i>.</p> <p><i>mr</i></p> <p>(Filled by the method) The transformation matrix.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewDrawCosObj AVPageViewDrawCosObjEx AVPageViewTransformRectRZ |

Example

```
ASFixedMatrix mr;
ASFixedRect ar;
CosObj aprObj;
AVRect avr;
ASUns32 flags = PDAnnotGetFlags(annot);

/* Draw an annotation appearance */
AVPageViewAppearanceGetAVMatrix(pageView,
    flags, aprObj, &ar, &mr);
AVPageViewDrawCosObjEx(pageView, aprObj,
    &avr, &mr);
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVPageViewBeginOperation

```
void AVPageViewBeginOperation (AVPageView pageView);
```

Description Increments an internal variable. Neither drawing nor [AVPageViewDidChange](#) notifications will occur as long as the variable has a value greater than zero. In addition, frames are not pushed onto the view history stack.

Parameters *pageView*
The page view whose *SuspendDraw* variable is incremented.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewEndOperation](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewDevicePointToPage

```
void AVPageViewDevicePointToPage (AVPageView pageView,  
    ASInt16 x, ASInt16 y, ASFixedPoint* p);
```

| | |
|------------------------|--|
| Description | Transforms a point's coordinates from device space to user space. |
| Parameters | <p><i>pageView</i></p> <p>Page view for which the point's coordinates are transformed.</p> <p><i>x</i></p> <p><i>x</i>-coordinate of the point to transform, specified in device space coordinates.</p> <p><i>y</i></p> <p><i>y</i>-coordinate of the point to transform, specified in device space coordinates.</p> <p><i>p</i></p> <p><i>(Filled by the method)</i> Pointer to a point whose user space coordinates correspond to <i>x</i> and <i>y</i>.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewPointToDevice AVPageViewRectToDevice AVPageViewDeviceRectToPage |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewDeviceRectToPage

```
void AVPageViewDeviceRectToPage (AVPageView pageView,
    const AVRect* rect, ASFixedRect* p);
```

Description Transforms a rectangle from device space to user space coordinates. The resulting *ASFixedRect* is “normal,” that is, *left* < *right* and *bottom* < *top*.

Parameters

pageView
Page view for which the rectangle is transformed.

rect
Pointer to a device space rectangle whose coordinates are transformed to user space.

p
(Filled by the method) Pointer to a user space rectangle corresponding to *rect*.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewPointToDevice](#)
[AVPageViewDevicePointToPage](#)
[AVPageViewDeviceRectToPageRZ](#)
[AVPageViewRectToDevice](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewDeviceRectToPageRZ

```
void AVPageViewDeviceRectToPageRZ (AVPageView pageView,
    ASInt32 flags, ASInt16 xHot, ASInt16 yHot,
    const AVRect* src, ASFixedRect* dest);
```

Description

Transforms an annotation's rectangle from device space to user space coordinates, allowing for the annotation's attributes of whether it should zoom or rotate when the page is zoomed or rotated. It also specifies a point that can remain in the view.

Parameters

pageView

Page view for which the rectangle is transformed.

flags

Flags to indicate whether the annotation rotates or zooms with the page view. These flags correspond to the annotation's **F** key and can be obtained from [PDAnnotGetFlags](#). Must be an OR of the following flags:

- *pdAnnotNoZoom*—Annotation does not zoom with the view.
- *pdAnnotNoRotate*—Annotation does not rotate with the page.

xHot

x-coordinate of point that should remain in the view.

yHot

y-coordinate of point that should remain in the view.

src

Pointer to a device space annotation rectangle whose coordinates are transformed to user space.

dest

(Filled by the method) Pointer to a user space rectangle corresponding to *src*.

Return Value

None

Exceptions

None

Notifications

None

Header File *AVCalls.h*

Related Methods [AVPageViewAppearanceGetAVMatrix](#)
[AVPageViewDeviceRectToPage](#)
[AVPageViewRectToDevice](#)
[AVPageViewTransformRectRZ](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVPageViewDoPopupMenu

```
AVMenuItem AVPageViewDoPopupMenu (AVPageView pageView,  
    AVMenu menu, ASInt16 xHit, ASInt16 yHit, ASBool rightMouse,  
    ASInt32 choice);
```

Description Displays the given *AVMenu* as a popup menu anchored at *xHit* and *yHit*, which are in device coordinates relative to *pageView*.

Parameters *pageView*

The page view in which the menu appears.

menu

The displayed menu.

xHit

The x-coordinate of the upper left corner of the menu.

yHit

The y-coordinate of the upper left corner of the menu.

rightMouse

true if the right mouse button (where applicable) was used to invoke the popup, *false* otherwise.

choice

The index of the *AVMenuItem* that should appear under the mouse at pop-up time.

Return Value The menu item selected from *menu*.

Header File *AVCalls.h*

Related Methods [AVToolButtonGetMenu](#)
[AVToolButtonSetMenu](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVPageViewDragOutNewRect

```
void AVPageViewDragOutNewRect (AVPageView pageView,  
    ASInt16 xStart, ASInt16 yStart, AVRect* resultRect);
```

| | |
|------------------------|--|
| Description | Allows the user to drag out a new rectangle. Call this method when the user clicks at some point and needs to create a rectangle. The method returns when the user releases the mouse button. |
| Parameters | <p><i>pageView</i></p> <p>The page view in which the rectangle is created.</p> <p><i>xStart</i></p> <p>The x-coordinate of the point where the user initially clicked, specified in device space coordinates.</p> <p><i>yStart</i></p> <p>The x-coordinate of the point where the user initially clicked, specified in device space coordinates.</p> <p><i>resultRect</i></p> <p>(Filled by the method) Pointer to the rectangle that the user dragged out, specified in device space coordinates.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewDragRect |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewDragRect

```
void AVPageViewDragRect (AVPageView pageView, ASInt16 xStart,
    ASInt16 yStart, AVRect* startRect, AVRect* resultRect,
    ASInt32 dragType, AVRect* extrema);
```

Description Allows the user to move or resize a rectangle. Call this method then the user clicks on a rectangle to modify. It returns after the user releases the mouse button.

Parameters *pageView*

The page view in which the rectangle is located.

xStart

The x-coordinate of the point where the user initially clicked, specified in device space coordinates.

yStart

The y-coordinate of the point where the user initially clicked, specified in device space coordinates.

startRect

Pointer to the initial rectangle, which is to moved or resized, specified in device space coordinates.

resultRect

(Filled by the method) Pointer to the resized or moved rectangle, specified in device space coordinates.

dragType

How the rectangle is changed. Must be a value between 0 and 4, inclusive. The values indicate:

0 — Move the rectangle.

1 — Resize the rectangle, using the lower right corner as an anchor point.

2 — Resize the rectangle, using the lower left corner as an anchor point.

3 — Resize the rectangle, using the upper left corner as an anchor point.

4 — Resize the rectangle, using the upper right corner as an anchor point.

extrema

Pointer to the rectangle specifying the maximum limits of the user-dragged rectangle. The user cannot grow the rectangle outside *extrema*, specified in device space coordinates. Pass *NULL* if you do not wish to limit the changes the user is making. *extrema* is ignored if *dragType* is 0.

| | |
|------------------------|---|
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewDeviceRectToPageRZ |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewDrawAnnotSequence

```
void AVPageViewDrawAnnotSequence (AVPageView pv, PDAnnot an,
    AVRect* bbox);
```

| | |
|------------------------|---|
| Description | Draws the annotation sequence number of an annotation in a rectangle. |
| Parameters | <p><i>pv</i></p> <p>The page view in which the annotation is drawn.</p> <p><i>an</i></p> <p>The annotation whose sequence number is drawn.</p> <p><i>bbox</i></p> <p>Bounding box in which to draw the sequence number of the annotation.</p> |
| Return Value | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewDrawRect |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVPageViewDrawCosObj

```
void AVPageViewDrawCosObj (AVPageView pageView, CosObj cosObj,
    AVRect* rect);
```

Description Draws the *CosObj* (which currently must be a Form object) and scales it to fit *rect*. This method may be used to draw an annotation appearance.

Parameters

pageView
The page view.

cosObj
The *CosObj* to draw.

rect
The rectangle in which to draw.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewAppearanceGetAVMatrix](#)
[AVPageViewDrawCosObjEx](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewDrawCosObjEx

```
void AVPageViewDrawCosObjEx (AVPageView pageView,  
    CosObj cosObj, AVRect* rect, ASFixedMatrix* matrix);
```

| | |
|------------------------|---|
| Description | <p>Draws the <i>CosObj</i> (which currently must be a Form object), scales it to fit <i>rect</i>, and transforms it according to <i>matrix</i>. This method is the same as AVPageViewDrawCosObj but applies a <i>ASFixedMatrix</i> transformation to the <i>CosObj</i> once the <i>CosObj</i> has been anamorphically scaled to the dimensions of the <i>AVRect</i>. This method may be used to draw an annotation appearance.</p> |
| Parameters | <p><i>pageView</i></p> <p>The page view.</p> <p><i>cosObj</i></p> <p>The <i>CosObj</i> to draw.</p> <p><i>rect</i></p> <p>The rectangle in which to draw.</p> <p><i>matrix</i></p> <p>Pointer to a matrix specifying the matrix to transform the <i>cosObj</i> based on <i>rect</i>. For example, if the <i>AVRect</i> is [10 20 110 220] describing a 100 x 200 area near the top left corner of the page, a <i>ASFixedMatrix</i> of [0.5 0 0 0.5 50 100] would scale the <i>CosObj</i> to 50% and center it inside the <i>AVRect</i>.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewAppearanceGetAVMatrix AVPageViewDrawCosObj |

Example

```
ASFixedMatrix mr;
ASFixedRect ar;
CosObj aprObj;
AVRect avr;
ASUns32 flags = PDAnnotGetFlags(annot);

/* Draw an annotation appearance */
AVPageViewAppearanceGetAVMatrix(pageView,
    flags, aprObj, &ar, &mr);
AVPageViewDrawCosObjEx(pageView, aprObj,
    &avr, &mr);
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewDrawNow

```
void AVPageViewDrawNow (AVPageView pageView);
```

Description Forces any pending updates for the specified page view to finish drawing.

Parameters *pageView*
The *AVPageView* to redraw.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewInvalidateRect](#)

Example

```
PDPage page;
AVRect avRect;
ASInt32 annotNum;
PDAnnot anAnnot;

for (annotNum = 0; annotNum <
    PDPageGetNumAnnots(page); annotNum++) {
    anAnnot = PDPageGetAnnot (page,
        annotNum);
    if (PDAnnotGetSubtype(anAnnot) ==
        ASAtomFromString("Text")) {
        AVPageViewGetAnnotRect(pageView,
            anAnnot, &avRect);
        AVPageViewInvalidateRect(pageView,
            &avRect);
    }
}
PDPageRelease(page);
AVPageViewDrawNow(pageView);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewDrawRect

```
void AVPageViewDrawRect (AVPageView pageView,
    const AVRect* rect);
```

Description Draws a rectangle filled with the color most recently set using [AVPageViewSetColor](#).

Parameters

pageView
The page view in which the rectangle is drawn.

rect
Pointer to the rectangle to draw, specified in device space coordinates.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewDrawRectOutline](#)
[AVPageViewDrawAnnotSequence](#)

Example

```
AVRect deviceRect;
AVDoc myAVDoc;
AVPageView currentPageView;

/* Set rectangle s coordinates */
deviceRect.top = 0;
deviceRect.left = 40;
deviceRect.right=80;
deviceRect.bottom=50;

/* Determine the current page view */
myAVDoc = AVAppGetActiveDoc();
if (myAVDoc != NULL) {
    currentPageView =
        AVDocGetPageView(myAVDoc);

    /* Draw the rectangle */
    AVPageViewDrawRect(currentPageView,
        &deviceRect);
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewDrawRectOutline

```
void AVPageViewDrawRectOutline (AVPageView pageView,  
    const AVRect* rect, ASInt16 lineWidth, ASFixed* dashArray,  
    ASInt32 arrayLen);
```

| | |
|------------------------|---|
| Description | Draws a stroked, but not filled, rectangle using the color most recently set using AVPageViewSetColor . |
| Parameters | <p><i>pageView</i></p> <p>The page view in which the rectangle is drawn.</p> <p><i>rect</i></p> <p>Pointer to the rectangle to draw, specified in device space coordinates.</p> <p><i>lineWidth</i></p> <p>Border width in pixels. For <i>lineWidth</i> > 1, the border line is entirely <i>inside</i> the <i>rect</i>, thus shrinking the area inside the outline.</p> <p><i>dashArray</i></p> <p>Pointer to an array of fixed numbers, whose elements alternately specify the length of dashes and gaps. Pass <i>NULL</i> to draw a solid outline.</p> <p><i>arrayLen</i></p> <p>Number of elements in a <i>dashArray</i>. Ignored if <i>dashArray</i> is <i>NULL</i>. The maximum allowed number of elements is currently 10.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewDrawRect |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewEndOperation

```
void AVPageViewEndOperation (AVPageView pageView);
```

| | |
|------------------------|--|
| Description | Decrements an internal variable. Neither drawing nor AVPageViewDidChange notifications will occur as long as the variable has a value greater than zero. In addition, frames are not pushed onto the view history stack. |
| Parameters | <p><i>pageView</i></p> <p>The page view whose <i>SuspendDraw</i> variable is decremented.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | AVPageViewDidChange |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewAppearanceGetAVMatrix |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetActiveBead

[PDBead](#) AVPageViewGetActiveBead ([AVPageView](#) pageView);

| | |
|------------------------|---|
| Description | Gets the currently-active article thread bead in the specified page view. |
| Parameters | <i>pageView</i> The page view whose currently-active bead is obtained. |
| Return Value | The current bead of the current thread, or a <i>NULL</i> Cos object if there is no current thread. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewGetThreadIndex |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetAnnotRect

```
void AVPageViewGetAnnotRect (AVPageView pageView,  
    PDAnnot anAnnot, AVRect* rect);
```

| | |
|------------------------|---|
| Description | Gets an annotation's bounding rectangle. This may be a superset of the actual bounding box of a particular annotation view. |
| Parameters | <p><i>pageView</i></p> <p>The page view for which the rectangle is transformed.</p> <p><i>anAnnot</i></p> <p>The annotation whose bounding rectangle is obtained.</p> <p><i>rect</i></p> <p>(Filled by the method) Pointer to the annotation's bounding rectangle, specified in device space coordinates.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewTransformRectRZ PDPageEnumResources |

Example

```

PDPPage page;
AVRect avRect;
ASInt32 annotNum;
PDAnnot anAnnot;

for (annotNum = 0; annotNum <
    PDPPageGetNumAnnots(page); annotNum++) {
    anAnnot = PDPPageGetAnnot
        (page,annotNum);
    if (PDAnnotGetSubtype(anAnnot) ==
        ASAtomFromString("Text")) {
        AVPageViewGetAnnotRect(pageView,
            anAnnot, &avRect);
        AVPageViewInvalidateRect(pageView,
            &avRect);
    }
}
PDPPageRelease(page);
AVPageViewDrawNow(pageView);

```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetAperture

```
void AVPageViewGetAperture (AVPageView pageView, AVRect* rect);
```

Description Gets the aperture of the specified page view. The aperture is the rectangular region of the window in which the document is drawn, measured in device space units.

Parameters

pageView
The page view whose aperture is obtained.

rect
(Filled by the method) Pointer to the aperture rectangle, specified in device space coordinates.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewInvalidateRect](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetAVDoc

[AVDoc](#) AVPageViewGetAVDoc ([AVPageView](#) pageView);

| | |
|------------------------|---|
| Description | Gets the <i>AVDoc</i> for the document currently displayed in <i>pageView</i> . |
| Parameters | <i>pageView</i> The page view whose <i>AVDoc</i> is obtained. |
| Return Value | The <i>AVDoc</i> for <i>pageView</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocGetPageView AVPageViewGetPage |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetColor

```
void AVPageViewGetColor (AVPageView pageView,  
    PDColorValue color);
```

Description Gets the color that will be used for subsequent drawing by [AVPageViewDrawRect](#) and [AVPageViewDrawRectOutline](#).

Parameters

pageView
The page view whose drawing color is obtained.

color
(Filled by the method) The color to get.

Return Value None

Exceptions [genErrBadParm](#)

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewGetColor](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetDevToPageMatrix

```
void AVPageViewGetDevToPageMatrix (AVPageView pageView,  
    ASFixedMatrix* devToPageMatrix);
```

| | |
|------------------------|--|
| Description | Gets the matrix that transforms device space coordinates to user space coordinates for the specified page view. |
| Parameters | <i>pageView</i> The page view whose matrix is obtained. <i>devToPageMatrix</i> (Filled by the method) Pointer to the transformation matrix. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewGetPageToDevMatrix |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetFirstVisiblePageNum

```
ASInt32 AVPageViewGetFirstVisiblePageNum (AVPageView pageView);
```

| | |
|------------------------|--|
| Description | Returns the page number of the first page that is visible on the screen. |
| Parameters | <p><i>pageView</i></p> <p>The page view.</p> |
| Return Value | Page number of the first page that is visible on the screen. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewGetLastVisiblePageNum AVPageViewPageNumIsVisible AVPageViewSetPageNum AVPageViewGetPageNum AVPageViewGetSelectedAnnotPageNum |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetGrayRect

```
void AVPageViewGetGrayRect (AVPageView pageView,  
    AVRect* greyRect);
```

Description Gets the rectangle that bounds a page view. This may include some of the “gray area” outside a page’s crop box.

Parameters

pageView
The page view.

greyRect
(Filled by the method) Rectangle bounding the page view, specified in device space coordinates.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewDrawRect](#)
[AVPageViewGetAperture](#)

Example

```
AVRect bounds;  
  
AVPageViewGetGratRect(pageView, &bounds);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVPageViewGetLastVisiblePageNum

```
ASInt32 AVPageViewGetLastVisiblePageNum (AVPageView pageView);
```

| | |
|------------------------|---|
| Description | Returns the page number of the last page that is visible on the screen. |
| Parameters | <p><i>pageView</i></p> <p>The page view.</p> |
| Return Value | Page number of the last page that is visible on the screen. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewGetFirstVisiblePageNum AVPageViewPageNumIsVisible AVPageViewSetPageNum AVPageViewGetPageNum AVPageViewGetSelectedAnnotPageNum |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetLayoutMode

[PDLayoutMode](#) AVPageViewGetLayoutMode ([AVPageView](#) pageView);

| | |
|------------------------|---|
| Description | Gets the page layout mode for a page view. |
| Parameters | <i>pageView</i> The page view whose layout mode is obtained. |
| Return Value | <i>pageView's</i> page layout mode, described in PDLayoutMode . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewSetLayoutMode |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetMousePosition

```
void AVPageViewGetMousePosition (AVPageView pageView,
    ASInt16* x, ASInt16* y);
```

| | |
|------------------------|---|
| Description | Gets the mouse position in <i>pageView</i> . The mouse position is specified in global coordinates (that is, in Mac OS, it is equivalent to calling the Toolbox <i>GetMouse</i> call and adding the window's offset on the screen; in UNIX, it is equivalent to calling <i>XQueryPointer</i> and adding the window's offset on the screen). |
| Parameters | <p><i>pageView</i></p> <p>The page view in which the mouse location is obtained.</p> <p><i>x</i></p> <p>(Filled by the method) The <i>x</i>-coordinate of the mouse location.</p> <p><i>y</i></p> <p>(Filled by the method) The <i>y</i>-coordinate of the mouse location.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVSysMouseIsStillDown |
| Example | <pre>AVPageViewGetMousePosition(pageView, &xHit, &yHit); if (AVPageViewIsAnnotAtPoint(pageView, xHit, yHit, &anAnnot)){ /* Pass the click to the annotation, causing the annotation s draw method to call */ }</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetNextView

```
ASBool AVPageViewGetNextView (AVPageView pageView,  
    PDDoc* pdDoc, ASInt32* pageNum, ASFixed* scale);
```

Description Used by page caching code to determine the next page to cache (and at what scale). Allows plug-ins to do their own page caching.

You can replace this method with your own version, using [HFTReplaceEntry](#).

Parameters

pageView
The page view.

pdDoc
(Filled by the method) The *PDDoc* containing the next page to cache.

pageNum
(Filled by the method) The next page to cache.

scale
(Filled by the method) The scale of the next page to cache.

Return Value *true* if the next page was cached, *false* otherwise.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVDocGetPageView](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetPage

[PDPPage](#) AVPageViewGetPage ([AVPageView](#) pageView);

| | |
|------------------------|--|
| Description | Gets the <i>PDPPage</i> currently displayed in the specified page view. This does <i>not</i> acquire the page. Do not use this result across calls that might change the current page. To obtain a value that can be used across such calls, use PDDocAcquirePage instead. |
| Parameters | <p><i>pageView</i></p> <p>The page view whose <i>PDPPage</i> is obtained.</p> |
| Return Value | <i>PDPPage</i> currently displayed in <i>pageView</i> , or <i>NULL</i> if there is not a valid <i>PDPPage</i> associated with <i>pageView</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | PDDocAcquirePage |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetPageNum

```
ASInt32 AVPageViewGetPageNum ( AVPageView pageView );
```

Description

Gets the current page number for *pageView*.

Note: If more than one page may be visible, use [AVPageViewGetFirstVisiblePageNum](#), [AVPageViewGetLastVisiblePageNum](#), or [AVPageViewPageNumIsVisible](#) instead of this method.

Parameters

pageView

The page view whose current page number is obtained.

Return Value

Current page number, or -1 if *pageView* is invalid. The first page in a document is page 0.

Exceptions

None

Notifications

None

Header File

AVCalls.h

Related Methods

[AVPageViewGetFirstVisiblePageNum](#)
[AVPageViewGetLastVisiblePageNum](#)
[AVPageViewPageNumIsVisible](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetPageToDevMatrix

```
void AVPageViewGetPageToDevMatrix (AVPageView pageView,  
    ASFixedMatrix* pageToDevMatrix);
```

| | |
|------------------------|---|
| Description | Gets the matrix that transforms user space coordinates to device space coordinates for the specified page view. |
| Parameters | <i>pageView</i> The page view whose transformation matrix is obtained. <i>pageToDevMatrix</i> (Filled by the method) Pointer to the transformation matrix. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | AVCalls.h |
| Related Methods | AVPageViewGetDevToPageMatrix |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetSelectedAnnotPageNum

```
ASInt32 AVPageViewGetSelectedAnnotPageNum
(AVPageView pageView);
```

| | |
|------------------------|---|
| Description | Returns the page number of the currently selected annotation or -1 if no annotation is selected. |
| Parameters | <i>pageView</i> The page view. |
| Return Value | Returns the page number of the currently selected annotation or -1 if no annotation is selected. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewGetFirstVisiblePageNum AVPageViewGetLastVisiblePageNum AVPageViewPageNumIsVisible AVPageViewSetPageNum AVPageViewGetPageNum |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetThreadIndex

```
ASInt32 AVPageViewGetThreadIndex (AVPageView pageView);
```

| | |
|------------------------|---|
| Description | Gets the index of the currently active thread in a page view. |
| Parameters | <p><i>pageView</i></p> <p>The page view whose active thread index is obtained.</p> |
| Return Value | The thread index of the current thread, or -1 if there is no current thread. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewGetActiveBead |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetVisibleAnnotPage

```
ASInt32 AVPageViewGetVisibleAnnotPage (AVPageView pageView,  
    PDAnnot annot);
```

Description Gets the number of a page containing an annotation.

Parameters *pageView*

The page view with the annotation.

annot

The annotation whose page number is sought.

Return Value The number of the page containing *annot*.

Header File *AVCalls.h*

Related Methods [AVPageViewGetFirstVisiblePageNum](#)
[AVPageViewGetLastVisiblePageNum](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVPageViewGetZoom

[ASFixed](#) AVPageViewGetZoom ([AVPageView](#) *pageView*);

| | |
|------------------------|---|
| Description | Gets the current zoom for <i>pageView</i> . |
| Parameters | <p><i>pageView</i></p> <p>The page view whose zoom is obtained.</p> |
| Return Value | Current zoom, as a fixed number measured in units in which 1.0 is 100% zoom. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewGetZoomType |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGetZoomType

[AVZoomType](#) AVPageViewGetZoomType ([AVPageView](#) pageView);

| | |
|------------------------|---|
| Description | Gets the current zoom type. |
| Parameters | <i>pageView</i> The page view whose zoom type is obtained. |
| Return Value | The current zoom type. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewGetZoom |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGoBack

```
void AVPageViewGoBack (AVPageView pageView);
```

| | |
|------------------------|--|
| Description | Goes to the previous view on the view stack, if a previous view exists. This might result in a different document being made active. |
| Parameters | <i>pageView</i> The page view to change. |
| Return Value | None |
| Exceptions | None |
| Notifications | AVPageViewDidChange |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewGoForward |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGoForward

```
void AVPageViewGoForward (AVPageView pageView);
```

| | |
|------------------------|--|
| Description | Goes to the next view on the view stack, if a next view exists. This might result in a different document being made active. |
| Parameters | <i>pageView</i> The page view to change. |
| Return Value | None |
| Exceptions | None |
| Notifications | AVPageViewDidChange |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewGoBack |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewGoTo

```
void AVPageViewGoTo (AVPageView pageView, ASInt32 pageNum);
```

Description Goes to specified page, retaining the current location on the page and the current zoom (either explicit or a variable). Invalidates the display, but does not perform an immediate redraw. This allows your plug-in to call [AVPageViewZoomTo](#), [AVPageViewScrollTo](#), or both and get only a single redraw event. If you decide to do this, you should bracket the calls with [AVPageViewAppearanceGetAVMatrix](#) and [AVPageViewEndOperation](#).

Parameters

pageView
The page view in which a different page is displayed.

pageNum
The page number of the destination page. The first page in a document is page 0.

Return Value None

Exceptions None

Notifications [AVPageViewDidChange](#)

Header File *AVCalls.h*

Related Methods [AVPageViewGetPage](#)
[AVPageViewGoBack](#)
[AVPageViewGoForward](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewHighlightText

```
void AVPageViewHighlightText ( AVPageView pageView,  
    PDTextSelect textSelect );
```

| | |
|------------------------|---|
| Description | Inverts the given text selection on the current page using the current <i>AVPageView</i> color. |
| Parameters | <i>pageView</i> The page view. <i>textSelect</i> The words to highlight. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewTrackText AVPageViewInvalidateText AVPageViewPointInText PDDocCreateStructTreeRoot |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewInsetRect

```
void AVPageViewInsetRect (AVPageView pageView,
    const AVRect* rect, ASBool down);
```

Description Draws a 3-D looking inset rectangle on the page view. Can be used to implement an “inset” style link where clicking the link causes the page to “push” into the screen.

Parameters

pageView
The page view on which to draw the rectangle.

rect
Rectangle to draw.

down
true to draw an inset rectangle, *false* to draw the rectangle in its normal state.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewDrawRect](#)
[AVPageViewInvertRect](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewInvalidateRect

```
void AVPageViewInvalidateRect (AVPageView pageView,  
    AVRect* area);
```

Description Indicates that the specified area of *pageView* is invalid and should be redrawn. This adds the rectangle to the list of regions to redraw, but does not force an immediate redraw. Use [AVPageViewDrawNow](#) to force an immediate redraw.

Parameters

pageView

The *AVPageView* in which a region is invalidated.

area

Pointer to the rectangle to invalidate, specified in device space coordinates. Use [AVPageViewRectToDevice](#) to convert a user space rectangle to device space. Pass *NULL* to invalidate the entire currently visible portion of the page.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Example

```
PDPage page;  
AVRect avRect;  
ASInt32 annotNum;  
PDAnnot anAnnot;  
  
for (annotNum = 0; annotNum <  
    PDPageGetNumAnnots(page); annotNum++) {  
    anAnnot = PDPageGetAnnot (page,  
        annotNum);  
    if (PDAnnotGetSubtype(anAnnot) ==  
        ASAtomFromString("Text")) {  
        AVPageViewGetAnnotRect(pageView,  
            anAnnot, &avRect);  
        AVPageViewInvalidateRect(pageView,  
            &avRect);  
    }  
}  
PDPageRelease(page);  
AVPageViewDrawNow(pageView);
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewInvalidateText

```
void AVPageViewInvalidateText (AVPageView pageView,  
    PDTextSelect textSelect);
```

| | |
|------------------------|---|
| Description | Invalidates the bits that AVPageViewHighlightText touches. |
| Parameters | <i>pageView</i> The page view. <i>textSelect</i> The words to invalidate. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewHighlightText AVPageViewPointInText AVPageViewTrackText |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewInvertQuad

```
void AVPageViewInvertQuad (AVPageView pageView,  
    const Quad* quad, ASBool highlight);
```

Description Inverts the interior of a quad.

Parameters *pageView*

The page view in which the inverted quad is drawn.

quad

Pointer to the quad to invert, specified in device space coordinates. Use [AVPageViewPointToDevice](#) to convert the coordinates of the four corners of the quad that is specified in user space.

highlight

If *true*, uses the highlight mode specified by *avpHighlightMode* in the Acrobat viewer's preferences file (see [AVAppSetPreference](#)). If *false*, uses a default highlighting mode.

Return Value None

Header File *AVCalls.h*

Related Methods [AVPageViewInvertRect](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVPageViewInvertRect

```
void AVPageViewInvertRect (AVPageView pageView,
    const AVRect* rect, ASBool highlight);
```

| | |
|------------------------|--|
| Description | Inverts the interior of a rectangle. |
| Parameters | <p><i>pageView</i></p> <p>The page view in which the inverted rectangle is drawn.</p> <p><i>rect</i></p> <p>Pointer to the rectangle to invert, specified in device space coordinates. Use AVPageViewRectToDevice to convert the coordinates of a rectangle that is specified in user space.</p> <p><i>highlight</i></p> <p>If <i>true</i>, uses the highlight mode specified by <i>avpHighlightMode</i> in the Acrobat viewer's preferences file (see AVAppSetPreference). If <i>false</i>, uses a default highlighting mode.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewDrawRect AVPageViewDrawRectOutline AVPageViewInsetRect AVPageViewInvertRectOutline |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

AVPageViewInvertRectOutline

```
void AVPageViewInvertRectOutline (AVPageView pageView,  
    const AVRect* rect);
```

| | |
|------------------------|--|
| Description | Inverts the specified rectangle's outline. |
| Parameters | <p><i>pageView</i></p> <p>The page view in which the inverted rectangle outline is drawn.</p> <p><i>rect</i></p> <p>Pointer to the rectangle whose outline is inverted, specified in device space coordinates. Use AVPageViewRectToDevice to convert the coordinates of a rectangle that is specified in user space coordinates.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewDrawRectOutline AVPageViewInvertRect AVSysMouselsStillDown |

Example

```
while (AVSysMouseIsStillDown())
{
    AVPageViewGetMousePosition(pageView,
        &x, &y);
    xDelta = x - xOld;
    yDelta = y - yOld;
    if (xDelta != 0 || yDelta != 0) {
        /* Remove old rectangle. */
        AVPageViewInvertRectOutline(pageView,
            &rr);
        /* Draw new rectangle reflecting the
            delta moved by the mouse.*/
        rr.left+= xDelta;
        rr.top+= yDelta;
        rr.right+= xDelta;
        rr.bottom+= yDelta;
        AVPageViewInvertRectOutline(pageView,
            &rr);

        /* Prepare for next move. */
        xOld = x;
        yOld = y;
    }
}
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewIsAnnotAtPoint

```
ASBool AVPageViewIsAnnotAtPoint (AVPageView pageView,  
    ASInt16 xHit, ASInt16 yHit, PDAnnot* hitAnnot);
```

| | |
|------------------------|---|
| Description | Tests whether or not the specified point is within an annotation. This method is typically used by mouse-handling code, to pass clicks within an annotation to the appropriate annotation handler. For each annotation, this method calls the appropriate annotation handler's AVAnnotHandlerPtInAnnotViewBBoxProc to test whether or not the point is within the annotation. |
| Parameters | <p><i>pageView</i> The page view for which the point to test.</p> <p><i>xHit</i> The x-coordinate of the point to test.</p> <p><i>yHit</i> The y-coordinate of the point to test.</p> <p><i>hitAnnot</i> (Filled by the method) Pointer to the topmost annotation (if any) that was hit by the mouse click.</p> |
| Return Value | <i>true</i> if the location specified by <i>xHit</i> and <i>yHit</i> is within an annotation, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppRegisterAnnotHandler AVPageViewGetAnnotRect AVPageViewGetMousePosition AVPageViewGetSelectedAnnotPageNum AVPageViewIsBeadAtPoint AVPageViewGetSelectedAnnotPageNum |

Example

```
if (AVPageViewIsAnnotAtPoint(pageView,
    xHit, yHit, &anAnnot)){
    /* Pass click on to the annotation to
       cause the annotation s draw function
       to call */
}
```

Availability

Plug-ins: Available if *PL_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewIsBeadAtPoint

```
ASBool AVPageViewIsBeadAtPoint (AVPageView pageView,  
    ASInt16 xHit, ASInt16 yHit, PDBead* beadP);
```

| | |
|----------------------|---|
| Description | Tests whether or not the specified point is within a bead. Returns the bead if it is. |
| Parameters | <p><i>pageView</i></p> <p>The page view in which the point is tested.</p> <p><i>xHit</i></p> <p>The x-coordinate of the point to test, specified in device space coordinates.</p> <p><i>yHit</i></p> <p>The y-coordinate of the point to test, specified in device space coordinates.</p> <p><i>beadP</i></p> <p>(Filled by the method) Pointer to the bead in which the point is located. This is only filled if the point is within a bead.</p> |
| Return Value | <i>true</i> if the point is within a bead, <i>false</i> otherwise. If the location is within a bead, the bead is returned in <i>beadP</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewPageNumIsVisible

```
ASBool AVPageViewPageNumIsVisible (AVPageView pageView,
    ASInt32 pageNum);
```

| | |
|------------------------|--|
| Description | Determines if a given page number is visible. |
| Parameters | <p><i>pageView</i></p> <p>The page view.</p> <p><i>pageNum</i></p> <p>The page number corresponding to the view of interest.</p> |
| Return Value | <i>true</i> if <i>pageNum</i> is visible, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewGetFirstVisiblePageNum AVPageViewGetLastVisiblePageNum AVPageViewGetPageNum AVPageViewGetSelectedAnnotPageNum AVPageViewSetPageNum |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewPointInText

```
ASBool AVPageViewPointInText (AVPageView pageView,
                               ASInt16 xHit, ASInt16 yHit, PDTextSelect pdText);
```

| | |
|------------------------|---|
| Description | Tests if the given point is in the <i>PDTextSelect</i> . |
| Parameters | <p><i>pageView</i> The page view.</p> <p><i>xHit</i> The x coordinate to test.</p> <p><i>yHit</i> The y coordinate to test.</p> <p><i>pdText</i> The text to hit-test upon.</p> |
| Return Value | <i>true</i> if the point is in the <i>textSelect</i> , <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewHighlightText AVPageViewInvalidateText AVPageViewTrackText |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewPointToDevice

```
void AVPageViewPointToDevice (AVPageView pageView,  
    const ASFixedPointP p, ASInt16* x, ASInt16* y);
```

| | |
|------------------------|--|
| Description | Transforms a point's coordinates from user space to device space. |
| Parameters | <p><i>pageView</i></p> <p>Page view for which the point's coordinates are transformed.</p> <p><i>p</i></p> <p>Pointer to the ASFixedPoint whose coordinates, specified in user space, are transformed.</p> <p><i>x</i></p> <p>(Filled by the method) <i>x</i>-coordinate of the device space point corresponding to <i>p</i>.</p> <p><i>y</i></p> <p>(Filled by the method) <i>y</i>-coordinate of the device space point corresponding to <i>p</i>.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewDevicePointToPage AVPageViewRectToDevice AVPageViewDeviceRectToPage |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewReadPageDown

```
void AVPageViewReadPageDown ( AVPageView pageView );
```

Description Scrolls down through a document, as if the user hit the Enter key. The scrolling follows articles if the Acrobat viewer is currently in article-reading mode.

Parameters *pageView*
The page view to scroll.

Return Value None

Exceptions None

Notifications [AVPageViewDidChange](#)

Header File *AVCalls.h*

Related Methods [AVPageViewReadPageUp](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewReadPageUp

```
void AVPageViewReadPageUp (AVPageView pageView);
```

| | |
|------------------------|---|
| Description | Scrolls up through a document, as if the user hit the Enter key. The scrolling follows articles if the Acrobat viewer is currently in article-reading mode. |
| Parameters | <i>pageView</i> The page view to scroll. |
| Return Value | None |
| Exceptions | None |
| Notifications | AVPageViewDidChange |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewReadPageDown |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewRectToDevice

```
void AVPageViewRectToDevice (AVPageView pageView,  
    const ASFixedRectP p, AVRect* rect);
```

Description Transforms a rectangle's coordinates from user space to device space. The resulting *AVRect* will be "normal," that is, *left* < *right* and *top* < *bottom*.

Parameters

pageView
Page view for which the coordinates are transformed.

p
Pointer to the rectangle whose coordinates are transformed, specified in user space coordinates.

rect
(Filled by the method) Pointer to a rectangle containing the device space coordinates corresponding to *p*.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewPointToDevice](#)
[AVPageViewDevicePointToPage](#)
[AVPageViewDeviceRectToPage](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewReleaseMachinePort

```
void AVPageViewReleaseMachinePort (AVPageView pageView,
    void* port);
```

| | |
|------------------------|---|
| Description | Releases the platform-specific object needed to draw into the Acrobat viewer's document window using a platform's native graphics calls. |
| Parameters | <p><i>pageView</i></p> <p>The <i>AVPageView</i> whose platform-dependent port is released.</p> <p><i>port</i></p> <p>The platform-specific port to release.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewAcquireMachinePort |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewScrollTo

```
void AVPageViewScrollTo (AVPageView pageView, ASInt16 xOrigin,  
    ASInt16 yOrigin);
```

Description Scrolls *pageView* to the location specified by *xOrigin* and *yOrigin*, within the limits imposed by the current zoom mode and the Acrobat viewer.

Parameters

pageView
The page view to scroll.

xOrigin
The x-coordinate to scroll to, specified in device space coordinates.

yOrigin
The y-coordinate to scroll to, specified in device space coordinates.

Return Value None

Exceptions None

Notifications [AVPageViewDidChange](#)

Header File *AVCalls.h*

Related Methods [AVPageViewScrollToRect](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewScrollToRect

```
void AVPageViewScrollToRect (AVPageView pageView,
    const AVRect* rect, ASBool favorLeft, ASBool favorTop,
    ASInt16 margin);
```

| | |
|------------------------|--|
| Description | Attempts to scroll the page view as little as possible to make the specified rectangle completely visible. This method is handy for auto-scrolling the <i>AVPageView</i> in a natural way to bring some page object completely into view. It will not affect the zoom level or AVZoomType . |
| Parameters | <p><i>pageView</i></p> <p>The page view to scroll.</p> <p><i>rect</i></p> <p>Pointer to the rectangle that is completely visible.</p> <p><i>favorLeft</i></p> <p>Used when <i>rect</i> is wider than the window's aperture. If <i>favorLeft</i> is <i>true</i>, favors the left side. If <i>false</i>, favors the right side. Favoring a side means that the corresponding edge will appear within the aperture, even if the opposite edge will not.</p> <p><i>favorTop</i></p> <p>Used when <i>rect</i> is taller than the window's aperture. If <i>favorTop</i> is <i>true</i>, favors the top side. If <i>false</i>, favors the bottom side. Favoring a side means that the corresponding edge will appear within the aperture, even if the opposite edge will not.</p> <p><i>margin</i></p> <p>Number of pixels that <i>rect</i> should be from the nearest edges if it doesn't cause the rectangle to go from completely visible to partially obscured.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | AVPageViewDidChange |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewScrollTo |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewSetAnnotLocation

```
void AVPageViewSetAnnotLocation (PDAnnot anAnnot,  
    AVPageView pageView, ASInt16 x, ASInt16 y);
```

| | |
|------------------------|--|
| Description | Sets an annotation's location, specified in device space coordinates. |
| Parameters | <p><i>anAnnot</i></p> <p>The annotation whose location is set.</p> <p><i>pageView</i></p> <p>The page view in which the annotation is displayed.</p> <p><i>x</i></p> <p>The annotation's new x-coordinate, specified in device space coordinates.</p> <p><i>y</i></p> <p>The annotation's new y-coordinate, specified in device space coordinates.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | PDAnnotWillChange PDAnnotDidChange |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewGetAnnotRect AVPageViewInvertQuad PDPageCreateAnnot |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewSetColor

```
void AVPageViewSetColor (AVPageView pageView,  
    PDColorValue color);
```

Description Sets the color that will be used for subsequent drawing by [AVPageViewDrawRect](#) and [AVPageViewDrawRectOutline](#).

Parameters

pageView
The page view whose drawing color is set.

color
The color to set.

Return Value None

Exceptions [genErrBadParm](#)

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewGetColor](#)

Example

```
PDColorValueRec red;  
    /* Define red */  
    red.space = PDDeviceRGB;  
    red.value[0] = Int32ToFixed(1);  
    red.value[1] = 0;  
    red.value[2] = 0;  
  
    /* set the drawing color */  
    AVPageViewSetColor(currentPageView,  
        &red);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewSetLayoutMode

```
void AVPageViewSetLayoutMode ( AVPageView pageView,  
    PDLayoutMode mode );
```

| | |
|------------------------|---|
| Description | Sets the layout mode for a page view. |
| Parameters | <i>pageView</i> The page view whose layout mode is set. <i>mode</i> The new layout mode for the page view. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewGetLayoutMode |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewSetPageNum

```
ASInt32 AVPageViewSetPageNum (AVPageView pageView,
    ASInt32 pageNum);
```

Description Sets the current logical page of the page view, which might not be the same as the current number indicated on the screen.

Plug-in writers in general do not need this, but if you wish to perform some operation on a page other than the one returned by [AVPageViewGetPageNum](#), you can use this call to temporarily set the page. You must restore the page number when you are done. You should avoid causing any major changes to the page view (such as scrolling) to ensure that you end up restoring the page to the correct value.

Parameters

pageView
The page view.

pageNum
The page number.

Return Value Previous page number.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewGetFirstVisiblePageNum](#)
[AVPageViewGetLastVisiblePageNum](#)
[AVPageViewGetPageNum](#)
[AVPageViewGetSelectedAnnotPageNum](#)
[AVPageViewPageNumIsVisible](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

AVPageViewShowControl

```
void AVPageViewShowControl (AVPageView pageView,  
    AVPageViewControllID controlID, ASBool show);
```

Description Turns on and off the controls shown in the status area at the bottom of a page view.

Parameters

pageView
The page view whose controls are affected.

controlID
The controls affected.

show
true if the controls are displayed, *false* if not shown.

Return Value None

Header File *AVCalls.h*

Related Methods None

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVPageViewStartReadingThread

```
void AVPageViewStartReadingThread ( AVPageView pageView,
    PDThread thread );
```

| | |
|------------------------|--|
| Description | Puts the specified page view into “article thread-reading” mode. |
| Parameters | <p><i>pageView</i></p> <p>The page view to set to article thread-reading mode.</p> <p><i>thread</i></p> <p>The thread to read.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewGetActiveBead |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewToDestInfo

[AVDestInfo](#) AVPageViewToDestInfo ([AVPageView](#) pageView,
[ASAtom](#) fitType);

| | |
|------------------------|---|
| Description | Creates a destination info object from a given <i>AVPageView</i> and <i>fitType</i> . |
| Parameters | <i>pageView</i> The page view from whose current view the destination info is created. <i>fitType</i> The <i>ASAtom</i> specifying the fit type that the view destination will have. The string associated with <i>fitType</i> must be one of View Destination Fit Types . |
| Return Value | The newly-created destination info. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDestInfoDestroy AVPageViewUseDestInfo |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020003 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewToViewDest

```
PDViewDestination AVPageViewToViewDest (AVPageView pageView,  
    ASAtom fitType, PDDoc srcPDDoc);
```

| | |
|------------------------|--|
| Description | Builds a <i>PDViewDestination</i> from the current zoom and position. |
| Parameters | <p><i>pageView</i></p> <p>The page view from whose current view the destination is created.</p> <p><i>fitType</i></p> <p>The <i>ASAtom</i> specifying the fit type that the view destination will have. The string associated with <i>fitType</i> must be one of View Destination Fit Types.</p> <p><i>srcPDDoc</i></p> <p>Document in which the view destination is used.</p> |
| Return Value | The newly-created view destination. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | PDViewDestCreate PDActionGetDest PDActionNewFromDest |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

AVPageViewTrackText

```
PDTextSelect AVPageViewTrackText (AVPageView pageView,  
ASInt16 xHit, ASInt16 yHit, PDTextSelect current);
```

Description Called in response to a mouse click to track a text selection on the screen. Uses the *AVPageView* current color, and leaves the screen with any highlights visible. Does not affect the current document selection.

Parameters

pageView
The page view.

xHit
The x-coordinate of the original click.

yHit
The y-coordinate of the original click.

current
Any existing selection that should be built upon.

Return Value *PDTextSelect* containing the words selected.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVPageViewHighlightText](#)
[AVPageViewInvalidateText](#)
[AVPageViewPointInText](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewTransformRectRZ

```
void AVPageViewTransformRectRZ (AVPageView pv, ASInt32 flags,
    ASFixedRect* ar, ASFixedMatrix* mr);
```

Description

Calculates where an annotation is drawn with no zoom or no rotation, as specified by the annotation flags.

This method generalizes [AVPageViewGetAnnotRect](#) and, except for zooming and rotating, is equivalent to the following:

```
PDAnnotGetRect(annot, &fr);
AVPageViewAnnotTransformRect(pv, annot,
    PDAnnotGetFlags(annot), &fr, &mr);
FixedRectRoundToRect16(&fr,
    AVRectToRect16(&ar));
AVPageViewGetAnnotRect(pv, annot, &ar);
```

Parameters

pv

The page view used for the transformation.

flags

Annotation flags obtained by *PDAnnotGetFlags*.

ar

Pointer to the annotation's bounding rectangle, specified in device space coordinates.

mr

(Filled by the method) The transformation matrix.

Return Value

None

Exceptions

None

Notifications

None

Header File

AVCalls.h

Related Methods

[AVPageViewAppearanceGetAVMatrix](#)
[AVPageViewDeviceRectToPageRZ](#)
[AVPageViewGetAnnotRect](#)

Example

```
ASFixedMatrix mr;
ASFixedRect ar, fr;
ASUns32 flags = PDAnnotGetFlags(annot);

PDAnnotGetRect(annot, &ar);
fr = ar;
AVPageViewTransformRectRZ(pageView, flags,
    &fr, &mr);
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVPageViewUseDestInfo

```
void AVPageViewUseDestInfo (AVPageView pageView,  
    AVDestInfo destInfo);
```

| | |
|------------------------|---|
| Description | Causes the given page view to change to the view given by an AVDestInfo object. |
| Parameters | <p><i>pageView</i> The page view to change.</p> <p><i>destInfo</i> The destination to use.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewToDestInfo AVPageViewUseThisDestination |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020003 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewUseThisDestination

```
void AVPageViewUseThisDestination (AVPageView pageView,
    PDViewDestination viewDest, ASFixed sourceZoom);
```

| | |
|------------------------|--|
| Description | Causes the given page view to change to the view given by <i>viewDest</i> and <i>sourceZoom</i> . |
| Parameters | <p><i>pageView</i> The page view to change.</p> <p><i>viewDest</i> The view destination.</p> <p><i>sourceZoom</i> The zoom factor to use, unless <i>viewDest</i> specifies inheriting the current zoom, which is the zoom in effect when a link is clicked, for example.</p> <p><i>sourceZoom</i> is used only if a) the view destination is of type XYZ (that is, specifies a point and a zoom) and b) the zoom is zero (meaning "inherit the current zoom"). In this case, <i>sourceZoom</i> is used as the zoom to inherit.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | PDViewDestCreate |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVPageViewZoomTo

```
void AVPageViewZoomTo (AVPageView pageView,  
    AVZoomType zoomType, ASFixed scale);
```

Description Sets the zoom factor and zoom type for the specified page view.

Parameters

pageView
The page view to zoom.

zoomType
The zoom type to set.

scale
Zoom factor, specified as a magnification factor (for example, 1.0 displays the document at actual size). *scale* is ignored unless *zoomType* is *AVZoomNoVary*. Use zero to inherit the zoom.

Return Value None

Exceptions None

Notifications [AVPageViewDidChange](#)

Header File *AVCalls.h*

Related Methods [AVPageViewScrollTo](#)
[AVPageViewScrollToRect](#)
[AVPageViewAppearanceGetAVMatrix](#)

Example

```
AVPageViewZoomTo (pageView, AVZoomFitPage,  
    fixedZero);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVSys

AVSysAllocTimeStringFromTimeRec

```
char* AVSysAllocTimeStringFromTimeRec (ASTimeRecP timeRec);
```

- Description** Given an [ASTimeRecP](#), gets a string representing the date and time. This routine is locale-friendly.
- The returned string is allocated using *ASmalloc* and must be freed by the caller with *ASfree*.
- Parameters** *timeRec*
- A pointer to an [ASTimeRec](#) structure.
- Return Value** String representing the date and time. Returns *NULL* if conversion fails.
- Header File** *AVCalls.h*
- Related Methods** None
- Availability** Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVSysBeep

```
void AVSysBeep (ASInt32 duration);
```

| | |
|------------------------|---|
| Description | Beeps. |
| Parameters | <i>duration</i> Always pass 0. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | None |
| Example | <code>AVSysBeep (0) ;</code> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVSysGetCursor

[AVCursor](#) AVSysGetCursor (void);

Description Gets the current cursor. Use this method when you want to change the cursor temporarily and be able to restore it to its current shape.

Parameters None

Return Value The current cursor.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVSysGetStandardCursor](#)
[AVSysSetCursor](#)

Example

```
oldCursor = AVSysGetCursor();
AVSysSetCursor(AVSysGetStandardCursor(
    WAIT_CURSOR));
/* do some work */

/* Return to old cursor */
AVSysSetCursor(oldCursor);
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVSysGetModifiers

```
ASUns32 AVSysGetModifiers (void);
```

| | |
|------------------------|---|
| Description | Gets a flag indicating which modifier keys are currently being pressed. |
| Parameters | None |
| Return Value | An OR of the Modifier Keys . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVSysMouselsStillDown |
| Example | <pre>ASBool shiftKeyIsDown = ((AVSysGetModifiers() & AV_SHIFT) != 0);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVSysGetStandardCursor

[AVCursor](#) AVSysGetStandardCursor (ASInt32 cursorID);

| | |
|------------------------|--|
| Description | Gets the specified cursor. The cursor can subsequently be displayed using AVSysSetCursor . |
| Parameters | <i>cursorID</i> The cursor to show. Must be one of the Predefined Cursors . |
| Return Value | The specified cursor. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVSysGetCursor AVSysSetCursor |
| Example | <pre>oldCursor = AVSysGetCursor(); AVSysSetCursor(AVSysGetStandardCursor(WAIT_CURSOR)); /* Do some work */ /* Return to old cursor */ AVSysSetCursor(oldCursor);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVSysMouseIsStillDown

```
ASBool AVSysMouseIsStillDown (void);
```

| | |
|------------------------|--|
| Description | Tests whether or not the mouse button is still being pressed. |
| Parameters | None |
| Return Value | <i>true</i> if the user is holding the mouse button down and hasn't released it since the last mouse-down event, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVSysGetModifiers |

Example

```
while (AVSysMouseIsStillDown())
{
    AVPageViewGetMousePosition(pageView,
        &x, &y);
    xDelta = x - xOld;
    yDelta = y - yOld;
    if (xDelta != 0 || yDelta != 0) {
        /* Remove old rectangle. */
        AVPageViewInvertRectOutline(pageView,
            &rr);
        /* Draw new rectangle reflecting the
           delta moved by the mouse */
        rr.left+= xDelta;
        rr.top+= yDelta;
        rr.right+= xDelta;
        rr.bottom+= yDelta;
        AVPageViewInvertRectOutline(pageView,
            &rr);

        /* Prepare for next move. */
        xOld = x;
        yOld = y;
    }
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVSysSetCursor

```
void AVSysSetCursor (AVCursor cursor);
```

| | |
|------------------------|---|
| Description | Sets the cursor to the specified <i>AVCursor</i> . |
| Parameters | <p><i>cursor</i></p> <p>The cursor to display. Predefined platform-independent cursors can be obtained using AVSysGetStandardCursor. Plug-ins can use their own custom cursors as follows:</p> <p>Macintosh users: Use the Macintosh Toolbox <i>GetCursor</i> call to retrieve your cursor from a resource. Cast the resulting <i>CursHandle</i> to an <i>AVCursor</i> and pass it to <i>AVSysSetCursor</i>.</p> <p>Windows users: Use the Windows API function <i>LoadCursor</i> to retrieve your cursor resource. Cast the resulting <i>HCURSOR</i> to an <i>AVCursor</i> and pass it to <i>AVSysSetCursor</i>.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVSysGetCursor AVSysGetStandardCursor |
| Example | <pre>oldCursor = AVSysGetCursor(); AVSysSetCursor(AVSysGetStandardCursor(WAIT_CURSOR)); /* Do some work */ /* Return to old cursor */ AVSysSetCursor(oldCursor);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVTool

AVToolGetType

[ASAtom](#) AVToolGetType ([AVTool](#) tool);

| | |
|------------------------|--|
| Description | Gets the currently active tools's type. See Tool Names for a list of the built-in tool types. |
| Parameters | <i>tool</i> The tool whose type is obtained. |
| Return Value | The ASAtom returned can be converted to a string using ASAtomGetString . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppEnumTools AVAppGetActiveTool AVAppGetDefaultTool AVAppGetLastActiveTool AVAppGetToolByName AVToolsPersistent |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolsPersistent

```
ASBool AVToolIsPersistent (AVTool tool);
```

Description Tests whether the specified tool is in a state keeping it active through multiple operations rather than only once, then restoring the previous tool. This method is called by another one-shot tool's *Activate* procedure. Two one-shot tools cannot cycle, because if the previous tool was not persistent, the second non-persistent tool reverts to the default tool.

Parameters *tool*
The tool whose persistence flag is tested.

Return Value *true* if the tool is persistent, *false* otherwise.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppEnumTools](#)
[AVAppGetActiveTool](#)
[AVAppGetDefaultTool](#)
[AVAppGetLastActiveTool](#)
[AVAppGetToolByName](#)
[AVToolGetType](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolBar

AVToolBarAddButton

```
void AVToolBarAddButton (AVToolBar toolBar, AVToolButton button,  
    ASBool before, AVToolButton otherButton);
```

| | |
|------------------------|--|
| Description | Inserts a button into a toolbar. Call AVToolBarUpdateButtonStates after adding a button to update the toolbar. |
| Parameters | <p><i>toolBar</i></p> <p>The toolbar into which a button is added.</p> <p><i>button</i></p> <p>The button to add to the toolbar.</p> <p><i>before</i></p> <p>If <i>otherButton</i> is <i>NULL</i> and <i>before</i> is <i>true</i>, the button is added to the left end of the toolbar. If <i>otherButton</i> is <i>NULL</i> and <i>before</i> is <i>false</i>, the button is added to the right end of the tool bar.</p> <p><i>otherButton</i></p> <p>A button relative to which the new button is added.</p> |
| Return Value | None |
| Exceptions | genErrNoMemory |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVToolButtonRemove |

Example

```
AVToolButton myButton;
AVToolbar mybar;

myButton = AVToolButtonNew(
    ASAtomFromString("Hello World"),
    GetResource('SICN', 600), FALSE, FALSE);
mybar = AVAppGetToolBar();
{
    AVToolButton oldButton =
        AVToolBarGetButtonByName(mybar,
            ASAtomFromString("EndDialogGroup"));
    AVToolBarAddButton(mybar, myButton,
        FALSE, oldButton);
}
AVToolBarGetNumButtons:
AVToolbar bar = AVAppGetToolBar();
if(AVToolBarGetNumButtons(bar)){

}
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolBarEnumButtons

```
void AVToolBarEnumButtons (AVToolBar toolBar,  
    AVToolButtonEnumProc enumProc, void* clientData);
```

| | |
|------------------------|---|
| Description | <p>Calls <i>enumProc</i> once for each toolbar button in the specified toolbar.</p> <p>If a toolbutton has a flyout, this is a <i>separate</i> toolbar from the toolbar returned by AVAppGetToolBar; enumerating the tool buttons on this main toolbar does <i>not</i> enumerate the tool buttons on any flyout. To enumerate the tool buttons on a flyout, call AVToolButtonGetFlyout to get its associated toolbar then call <i>AVToolBarEnumButtons</i> with this toolbar.</p> |
| Parameters | <p><i>toolBar</i></p> <p>The toolbar whose buttons are enumerated.</p> <p><i>enumProc</i></p> <p>User-supplied procedure to call once for each button.</p> <p>Enumeration ends if <i>enumProc</i> returns <i>false</i>.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>enumProc</i> each time it is called.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppGetToolBar AVToolButtonGetFlyout |

Example

```
/* Find the Hand toolbar button */
ACCB1 ASBool ACCB2 myButtonEnum
(AVToolButton button, void* clientData){
    if(AVToolButtonGetName(button) ==
        ASAtomFromString("Hand"))
        return false;
    return true;
}
AVToolBarEnumButtons(bar,
    ASCallbackCreateProto(
        AVToolButtonEnumProc, &myButtonEnum),
    NULL);
```

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolBarGetButtonByName

[AVToolButton](#) AVToolBarGetButtonByName ([AVToolBar](#) toolBar ,
[ASAtom](#) buttonName) ;

| | |
|----------------------|--|
| Description | Gets the toolbar button that has the specified name. |
| Parameters | <p><i>toolBar</i></p> <p>The toolbar in which the button is located.</p> <p><i>buttonName</i></p> <p>The <i>ASAtom</i> for the button to get. The character string representing <i>buttonName</i> can be converted to an <i>ASAtom</i> using ASAtomFromString. See Toolbar Button Names for a list of the names of the built-in buttons.</p> |
| Return Value | The button with the specified name. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Example | <pre>/* Add a button just before the endToolsGroup separator */ AVToolBarAddButton(AVAppGetToolBar() , myToolButton, true, AVToolBarGetButtonByName(ToolBar , ASAtomFromString("endToolsGroup"))) ;</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolBarGetFrame

```
void AVToolBarGetFrame (AVToolBar toolbar, AVRect* frame);
```

| | |
|------------------------|---|
| Description | Gets the coordinates of <i>toolbar</i> 's frame. UNIX users: The toolbar frame is not needed or used, and this method returns a meaningless frame. |
| Parameters | <p><i>toolbar</i></p> <p>The toolbar whose frame is obtained.</p> <p><i>frame</i></p> <p>(Filled by the method) Pointer to a rectangle specifying the toolbar's frame, specified in device space coordinates.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | AVCalls.h |
| Related Methods | AVAppGetToolBar |
| Example | <pre>AVRect frame;</pre> <pre>AVToolBarGetFrame(AVAppGetToolBar(),</pre> <pre>frame);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolBarGetNumButtons

```
ASInt32 AVToolBarGetNumButtons (AVToolBar toolbar);
```

| | |
|------------------------|---|
| Description | Gets the number of buttons in <i>toolbar</i> . |
| Parameters | <p><i>toolbar</i></p> <p>The toolbar whose button count is obtained.</p> |
| Return Value | The number of buttons in <i>toolbar</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppGetToolBar |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolBarIsRoomFor

```
ASBool AVToolBarIsRoomFor (AVToolBar toolbar, ASInt16 nButtons,
    ASInt16 nSeparators);
```

Description Tests whether or not there is room in a toolbar for an additional specified number of buttons and separators. In Windows, this method assumes the application window has been maximized.

Parameters

toolbar
The toolbar to check.

nButtons
The number of buttons.

nSeparators
The number of separators.

Return Value *true* if there is room in *toolbar* to add *nButtons* and *nSeparators*, *false* otherwise.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppGetToolBar](#)
[AVToolBarGetFrame](#)
[AVToolBarGetNumButtons](#)
[AVToolBarButtonNew](#)

Example

```
AVToolBar mybar = AVAppGetToolBar();
freeSlot = AVToolBarIsRoomFor(mybar, 1, 0);
if (freeSlot){
    /* add new button */
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolBarNewFlyout

[AVToolBar](#) AVToolBarNewFlyout (void);

Description

Creates a new sub-toolbar for use as a tool button *flyout*.

Flyouts are established by creating a new toolbar with *AVToolBarNewFlyout*, appending tool buttons to the new toolbar using *AVToolBar* calls, and attaching that toolbar to a toolbutton, known as the *anchor button*, with [AVToolButtonSetFlyout](#).

This method creates a distinct toolbar from the toolbar returned by [AVAppGetToolBar](#).

Note: The viewer makes a copy of the anchor button and attaches it to the front of the flyout to achieve the correct visual effect; the client doesn't need to do this.

Parameters

None

Return Value

The newly-created toolbar to use for a flyout.

Header File

AVCalls.h

Related Methods

[AVToolButtonGetFlyout](#)
[AVToolButtonSetFlyout](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVToolBarUpdateButtonStates

```
void AVToolBarUpdateButtonStates (AVToolBar toolbar);
```

Description Forces a redraw of *toolbar*. Call this method when a tool button is added or removed or one of the buttons changes state.

Parameters *toolbar*
The toolbar to redraw.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVAppGetToolBar](#)
[AVToolBarGetFrame](#)
[AVToolBarIsRoomFor](#)
[AVToolButtonNew](#)

Example

```
AVToolBarUpdateButtonStates(  
    AVAppGetToolBar() );
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolButton

AVToolButtonDestroy

```
void AVToolButtonDestroy (AVToolButton toolButton);
```

| | |
|------------------------|--|
| Description | Removes the specified button from the toolbar and destroys the button. Call AVToolBarUpdateButtonStates after removing a button to update the toolbar. |
| Parameters | <p><i>toolButton</i></p> <p>The button to destroy.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVToolButtonNew |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolButtonExecute

```
void AVToolButtonExecute (AVToolButton button);
```

| | |
|------------------------|---|
| Description | Executes the AVExecuteProc associated with <i>button</i> , if it exists. Does nothing if AVToolButtonIsEnabled (<i>button</i>) returns <i>false</i> . |
| Parameters | <i>button</i> The button whose execute proc is executed. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVToolButtonIsEnabled AVToolButtonNew AVToolButtonSetExecuteProc |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolButtonGetFlyout

[AVToolBar](#) AVToolButtonGetFlyout ([AVToolButton](#) button);

- Description** Gets the *flyout* attached to a tool button. A flyout is a sub-toolbar attached to a tool button.
- This method gets a *different* toolbar from the toolbar returned by [AVAppGetToolBar](#).
- Parameters** *button*
- The tool button for which the flyout is obtained.
- Return Value** The flyout associated with *button*. Returns *NULL* if there is no flyout associated with *button*.
- Header File** *AVCalls.h*
- Related Methods** [AVToolBarNewFlyout](#)
[AVToolButtonSetFlyout](#)
- Availability** Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVToolButtonGetIcon

[AVIcon](#) AVToolButtonGetIcon ([AVToolButton](#) button);

| | |
|------------------------|---|
| Description | Gets the icon associated with the specified <i>AVToolButton</i> . |
| Parameters | <p><i>button</i></p> <p>The button whose icon is returned.</p> |
| Return Value | The icon associated with <i>button</i> . |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVToolButtonSetIcon |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVToolButtonGetMenu

[AVMenu](#) AVToolButtonGetMenu ([AVToolButton](#) button);

| | |
|------------------------|---|
| Description | Obtains the menu attached to a tool button. |
| Parameters | <p><i>button</i></p> <p>The tool button to which a menu is attached.</p> |
| Return Value | The menu attached to <i>button</i> . |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVPageViewDoPopupMenu AVToolButtonSetMenu |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVToolButtonGetName

[ASAtom](#) AVToolButtonGetName ([AVToolButton](#) button);

| | |
|------------------------|--|
| Description | Gets the <i>ASAtom</i> corresponding to the name of the specified toolbar button. |
| Parameters | <i>button</i> The toolbar button whose name is obtained. |
| Return Value | The <i>ASAtom</i> corresponding to the toolbar button's name. <i>ASAtom</i> can be converted to a character string using ASAtomGetString . See Toolbar Button Names for a list of the built-in button names. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVToolBarGetButtonByName |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolButtonIsEnabled

```
ASBool AVToolButtonIsEnabled (AVToolButton button);
```

| | |
|------------------------|--|
| Description | Tests whether or not a toolbar button is enabled. |
| Parameters | <p><i>button</i></p> <p>The button to test.</p> |
| Return Value | <i>true</i> if <i>button</i> 's AVComputeEnabledProc returns <i>true</i> , <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVToolButtonSetComputeEnabledProc |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolButtonIsMarked

```
ASBool AVToolButtonIsMarked ( AVToolButton button );
```

| | |
|------------------------|--|
| Description | Tests whether or not the specified button is marked. |
| Parameters | <p><i>button</i></p> <p>The button to test.</p> |
| Return Value | <i>true</i> if <i>button</i> 's AVComputeMarkedProc returns <i>true</i> , <i>false</i> if <i>button</i> is not marked or doesn't have an AVComputeMarkedProc . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVToolButtonSetComputeMarkedProc |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolButtonIsSeparator

```
ASBool AVToolButtonIsSeparator (AVToolButton toolButton);
```

| | |
|------------------------|--|
| Description | Tests whether a toolbar button is a separator or a normal button. |
| Parameters | <i>button</i> The button to test. |
| Return Value | <i>true</i> if the button is a separator, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVToolButtonGetIcon AVToolButtonIsEnabled AVToolButtonIsMarked AVToolButtonSetComputeMarkedProc |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolButtonNew

[AVToolButton](#) AVToolButtonNew ([ASAtom](#) name, [AVIcon](#) icon, ASBool longOnly, ASBool isSeparator);

Description Creates a toolbar button with the specified name, icon and long-menus state. Can also be used to create a separator with the specified name.

Parameters

name

The *ASAtom* corresponding to the button's name. The character string for *name* can be converted to an *ASAtom* using [ASAtomFromString](#).

icon

The icon to use for this button.

In Mac OS, *icon* must be a handle to a standard *SICN* resource.

In Windows, *icon* is an 18×18 icon with a light gray background (that is, RGB values of 192,192,192).

longOnly

(Ignored in Acrobat 3.0 or later) If *true*, the button is shown only when the user selects "Full menus" in the Acrobat viewer. If *false*, shows in both "Full menu" and "Short menu" modes.

isSeparator

If *true*, the new button is a separator used to leave space between groups of related buttons. For separators, *icon*, the button's [AVExecuteProc](#), [AVComputeEnabledProc](#), and [AVComputeMarkedProc](#) are ignored. In addition, separators are not clickable.

If *false*, the button is a normal toolbar button.

Return Value The newly-created button.

Exceptions [genErrNoMemory](#)

Notifications None

Header File *AVCalls.h*

Related Methods [AVToolButtonDestroy](#)
[AVToolButtonSetExecuteProc](#)
[AVToolButtonSetComputeEnabledProc](#)
[AVToolButtonSetComputeMarkedProc](#)

Example

```
AVToolButton myButton;
AVToolBar mybar;

myButton = AVToolButtonNew(
    ASAtomFromString("Hello World"),
    GetResource('SICN', 600), FALSE, FALSE);
mybar = AVAppGetToolBar();
freeslots = AVToolBarIsRoomFor(mybar, 1,
    0);

if (freeslots){
    /* add new button after all others */
    AVToolButton oldButton =
        AVToolBarGetButtonByName(mybar,
            ASAtomFromString("EndDialogGroup"));
    AVToolBarAddButton(mybar, myButton,
        FALSE, oldButton);
    AVToolButtonSetExecuteProc(myButton,
        doHelloAlertCallback, NULL);
    AVToolButtonSetComputeEnabledProc(
        myButton, DocExistEnableCallback,
        NULL);
    AVToolButtonSetComputeMarkedProc(
        myButton, markedProcCallback, NULL);
}
```

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolButtonRemove

```
void AVToolButtonRemove (AVToolButton toolButton);
```

Description Removes the specified button from the toolbar, but does not destroy the button. Call [AVToolBarUpdateButtonStates](#) after removing a button to update the toolbar.

Parameters *toolButton*
The button to remove.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVToolBarAddButton](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolButtonSetComputeEnabledProc

```
void AVToolButtonSetComputeEnabledProc (AVToolButton button,
AVComputeEnabledProc proc, void* clientData);
```

| | |
|------------------------|--|
| Description | Sets the AVComputeEnabledProc associated with a toolbar button. This routine determines whether the button can be selected. |
| Parameters | <p><i>button</i></p> <p>The button whose AVComputeEnabledProc is set.</p> <p><i>proc</i></p> <p>User-supplied procedure to call whenever the Acrobat viewer needs to know whether or not <i>button</i> should be enabled.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>proc</i> each time it is called.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVToolButtonIsEnabled |
| Example | <pre>AVToolButton oldButton = AVToolBarGetButtonByName(mybar, ASAtomFromString("EndDialogGroup")); AVToolBarAddButton(mybar, myButton, FALSE, oldButton); AVToolButtonSetExecuteProc(myButton, doHelloAlertCallback, NULL); AVToolButtonSetComputeEnabledProc(myButton, DocExistEnableCallback, NULL); AVToolButtonSetComputeMarkedProc(myButton, markedProcCallback, NULL);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolButtonSetComputeMarkedProc

```
void AVToolButtonSetComputeMarkedProc (AVToolButton button,
AVComputeMarkedProc proc, void* clientData);
```

| | |
|------------------------|--|
| Description | Sets the AVComputeMarkedProc associated with a toolbar button. A marked button appears pressed on the screen. |
| Parameters | <p><i>button</i></p> <p>The button whose AVComputeMarkedProc is set.</p> <p><i>proc</i></p> <p>User-supplied callback to call whenever the Acrobat viewer needs to know whether or not the specified toolbar button should be marked.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>proc</i> each time it is called.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVToolButtonIsMarked AVToolButtonNew AVToolButtonSetComputeEnabledProc AVToolButtonSetExecuteProc |
| Example | <pre>AVToolButton oldButton = AVToolBarGetButtonByName(mybar, ASAtomFromString("EndDialogGroup")); AVToolBarAddButton(mybar, myButton, FALSE, oldButton); AVToolButtonSetExecuteProc(myButton, doHelloAlertCallback, NULL); AVToolButtonSetComputeEnabledProc(myButton, DocExistEnableCallback, NULL); AVToolButtonSetComputeMarkedProc(myButton, markedProcCallback, NULL);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolButtonSetExecuteProc

```
void AVToolButtonSetExecuteProc (AVToolButton button,
    AVExecuteProc proc, void* clientData);
```

| | |
|------------------------|--|
| Description | Sets the user-supplied procedure to call to actually “do” whatever the button does. |
| Parameters | <p><i>button</i></p> <p>The button whose AVExecuteProc is set.</p> <p><i>proc</i></p> <p>User-supplied procedure to call when <i>button</i> is executed.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>proc</i> each time it is called.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVToolButtonNew AVToolButtonSetComputeEnabledProc AVToolButtonSetComputeMarkedProc |
| Example | <pre>AVToolButton oldButton = AVToolBarGetButtonByName(mybar, ASAtomFromString("EndDialogGroup")); AVToolBarAddButton(mybar, myButton, FALSE, oldButton); AVToolButtonSetExecuteProc(myButton, doHelloAlertCallback, NULL); AVToolButtonSetComputeEnabledProc(myButton, DocExistEnableCallback, NULL); AVToolButtonSetComputeMarkedProc(myButton, markedProcCallback, NULL);</pre> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolButtonSetExternal

```
void AVToolButtonSetExternal (AVToolButton button,
    ASUns16 external);
```

Description

Indicates that the specified toolbar button should be displayed in toolbars contained in external windows, such as in a Web browser.

Note: Call AVToolButtonSetExternal before adding the toolbar button with [AVToolBarAddButton](#). If you want to change the toolbar button's location after it has been added, call AVToolButtonSetExternal and re-add the button with [AVToolBarAddButton](#).

Parameters

button

The button.

external

Indicates whether or not to show the button in external windows. Must be one of [Tool Button Flags](#).

Note: To enable a tool button in an external window, first remove the button from the tool bar, then turn on the TOOLBUTTON_EXTERNAL flag and add the button back on to the toolbar.

Return Value

None

Exceptions

None

Notifications

None

Header File

AVCalls.h

Related Methods

[AVToolButtonNew](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|-----------|------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

AVToolButtonSetFlyout

```
void AVToolButtonSetFlyout (AVToolButton button, AVToolBar bar);
```

Description

Attaches a sub-toolbar or *flyout* to a tool button. A copy of the button is attached to the front of the toolbar. Click-hold pops up the *flyout* and allow the user to select a different button.

Flyouts are established by creating a new toolbar with [AVToolBarNewFlyout](#), appending tool buttons to the new toolbar using *AVToolBar* calls, and attaching that toolbar to a toolbutton, known as the *anchor button*, with *AVToolButtonSetFlyout*.

Parameters

button

The tool button to attach to the flyout.

bar

The flyout to which *button* is attached.

Return Value

None

Header File

AVCalls.h

Related Methods

[AVToolBarNewFlyout](#)
[AVToolButtonGetFlyout](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVToolButtonSetHelpText

```
void AVToolButtonSetHelpText (AVToolButton button,
    const char* text);
```

Description

Sets the text to show in “tooltips.” This text is shown when the cursor is held over a tool button for period of time.

In Acrobat 4.0 and later, tool buttons can also have single key shortcuts assigned to them. (Conceptually, tools have shortcuts but we attach the shortcuts to the buttons.) This is done by appending a vertical bar character “|” followed by the shortcut to the button’s tooltip text. For example, here’s the tooltip text for the Hand tool:

“Hand Tool (H)|h”

The trailing “|h” portion indicates that the Hand tool uses the ‘H’ key as the shortcut. This portion is stripped off before the tooltip is displayed.

This behavior only applies to tooltips where the “|” is the second-to-last character.

Appending the shortcut to the tooltip text allows it to localize at the same time as the tooltip text.

Parameters

button

The tool button to which a tooltip is added.

text

The text to show.

Return Value

None

Exceptions

None

Notifications

None

Header File

AVCalls.h

Related Methods

[AVToolButtonNew](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVToolButtonSetIcon

```
void AVToolButtonSetIcon (AVToolButton button, AVIcon icon);
```

Description

Sets a new icon for a tool button.

A toolbutton's icon can change dynamically. This allows multi-state buttons, such as the one that opens and closes the splitter bar, to change appearance appropriately. This should allow multiple buttons to collapse into one.

Most other aspects of a toolbutton, such as execute proc and tooltip text, can be changed on the fly using other *AVToolButton* methods.

Parameters

button

The tool button whose icon is set.

icon

The icon to place on *button*.

Return Value

None

Header File

AVCalls.h

Related Methods

[AVToolButtonGetIcon](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVToolButtonSetMenu

```
void AVToolButtonSetMenu (AVToolButton button, AVMenu menu);
```

Description Attaches a menu to a tool button.

If a toolbutton has no execute proc, the menu pops up when the toolbutton is clicked. If the toolbutton does have an execute proc, the user must click and hold on the button for some time to display the menu. Simply clicking on the button invokes the button's execute proc as usual.

Parameters

button

The tool button to which a menu is attached.

menu

The menu to attach to *button*.

Return Value None

Header File *AVCalls.h*

Related Methods [AVPageViewDoPopupMenu](#)
[AVToolButtonGetMenu](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVWindow

AVWindowBecomeKey

```
void AVWindowBecomeKey (AVWindow win);
```

| | |
|------------------------|--|
| Description | Makes <i>win</i> the key window (regardless of the setting of its <i>WantsKey</i> flag) if the window is visible. UNIX users: This method is a no-op. |
| Parameters | <i>win</i> The window that is to become the key window. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVWindowSetWantsKey AVWindowIsVisible AVWindowResignKey |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowBringToFront

```
void AVWindowBringToFront (AVWindow win);
```

| | |
|------------------------|---|
| Description | Brings the specified window to the front. |
| Parameters | <p><i>win</i></p> <p>The window to bring to the front.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | AVAppFrontDocDidChange |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVWindowIsKey AVWindowSetWantsKey |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowDestroy

```
void AVWindowDestroy (AVWindow win);
```

Description Destroys the specified window and all associated memory. Closes the window without calling the window handler's [AVWindowWillCloseProc](#) (that is, this operation cannot be rejected).

Parameters *win*
The window to destroy.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowDrawNow

```
void AVWindowDrawNow (AVWindow win);
```

| | |
|------------------------|---|
| Description | Redraws the invalid regions of the specified window. |
| Parameters | <p><i>win</i></p> <p>The window to draw.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppRegisterForPageViewDrawing |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowGetFrame

```
void AVWindowGetFrame (AVWindow win, AVRect* rect);
```

Description

Gets the window's frame, which specifies its size and location on the screen.

Note: In Mac OS, this method may change the current port, thus altering the Macintosh graphics state. It sets the port to that specified by win, but fails to restore the previous port before exiting.

Parameters

win

The window whose frame is obtained.

rect

(Filled by the method) Pointer to a rectangle specifying the window's frame rectangle, specified in global screen coordinates.

In Mac OS, the frame includes only the window's content region.

In Windows, it includes the entire window (content region, title bar, and so forth).

Return Value

None

Exceptions

None

Notifications

None

Header File

AVCalls.h

Related Methods

[AVWindowSetFrame](#)
[AVWindowGetInterior](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowGetInterior

```
void AVWindowGetInterior (AVWindow win, AVRect* rect);
```

| | |
|------------------------|--|
| Description | Gets the interior rectangle of the window. |
| Parameters | <p><i>win</i></p> <p>The window whose interior is obtained.</p> <p><i>rect</i></p> <p>(Filled by the method) Pointer to a rectangle specifying the window's interior, specified as local window coordinates.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVWindowGetFrame AVWindowSetFrame |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowGetOwnerData

```
void* AVWindowGetOwnerData (AVWindow win);
```

Description Gets a window's owner data. The owner data is private data for the use of the window's creator. For example, if a plug-in uses its own class library, it might use the owner data field to store a pointer to the object owning the *AVWindow*.

Parameters

win
The window whose owner data is obtained.

newData
Pointer to the new owner data for *win*.

Return Value Pointer to owner data for *win*.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVWindowSetOwnerData](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowGetPlatformThing

```
void* AVWindowGetPlatformThing (AVWindow win);
```

| | |
|------------------------|---|
| Description | Returns a pointer to the platform-specific thing associated with the window. Do not confuse this with the owner data (see also AVWindowGetOwnerData). |
| Parameters | <p><i>win</i></p> <p>Platform-dependent window thing: a <i>WindowPtr</i> in Mac OS, an <i>HWND</i> in Windows, and a <i>Widget</i> in UNIX.</p> |
| Return Value | The platform-dependent thing for the window. <i>NULL</i> if the window is associated with an <i>AVDoc</i> that has been set dead by AVDocSetDead . |
| Exceptions | AVDocWantsToDie is broadcast after AVDocSetDead has been called, which would warn that the <i>AVWindow</i> associated with that window is <i>NULL</i> . |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVDocSetDead AVWindowNewFromPlatformThing AVWindowNew |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowGetTitle

```
ASInt32 AVWindowGetTitle (AVWindow win, char* buffer,
    ASInt32 bufferLen);
```

Description Gets the title to display in the specified window's title bar.

Parameters

win

The window whose title is obtained.

buffer

(Filled by the method) The buffer into which the window's title is read.

bufferLen

Length of *buffer*, in bytes.

Return Value The number of characters copied into *buffer*. If *buffer* is *NULL*, but *win* is not, the number of characters in the window title. If *win* is *NULL*, returns 0.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVWindowSetTitle](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowHandlePlatformEvent

```
ASBool AVWindowHandlePlatformEvent (AVWindow win,
    void* platformEvent);
```

| | |
|------------------------|--|
| Description | Handles a platform-specific event. Use this method to dispatch a platform-specific event structure to an <i>AVWindow</i> . |
| Parameters | <p><i>win</i></p> <p>The window with the event to handle.</p> <p><i>platformEvent</i></p> <p>A pointer to a platform-specific event structure.</p> |
| Return Value | <i>true</i> if the event was handled, <i>false</i> otherwise. |
| Exceptions | May raise exceptions, depending on the event. |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVAppHandleAppleEvent AVAppHandlePlatformEvent |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

AVWindowHide

```
void AVWindowHide (AVWindow win);
```

Description

Hides the specified window. Hiding a window makes it invisible, it does not minimize or iconize it.

In Windows, a document window can be minimized using code based on the following:

```
theAVWindow = AVDocGetAVWindow(theAVDoc);
theThing = (HWND) AVWindowGetPlatformThing(
theAVWindow);
wasVisible = ShowWindow(theThing,
SW_MINIMIZED);
```

Parameters

win

The window to hide.

Return Value

None

Exceptions

None

Notifications

None

Header File

AVCalls.h

Related Methods

[AVWindowShow](#)
[AVWindowGetPlatformThing](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowInvalidateRect

```
void AVWindowInvalidateRect (AVWindow win, const AVRect* rect);
```

Description Invalidates the specified rectangular region, for eventual redraw. This is the preferred method for refreshing a portion of an *AVWindow*. Use [AVWindowDrawNow](#) to force a window to redraw its invalid regions.

Parameters

win
The window in which to invalidate *rect*.

rect
Pointer to a rectangle specifying the region to invalidate.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVWindowDrawNow](#)
[AVWindowGetInterior](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowIsKey

```
ASBool AVWindowIsKey (AVWindow win);
```

Description Tests whether or not the specified window is the key window. The key window is the window that receives mouse clicks.
UNIX users: This method is a no-op.

Parameters *win*
The window to test.

Return Value *true* if *win* is the key window, *false* otherwise.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVWindowBecomeKey](#)
[AVWindowBringToFront](#)
[AVWindowSetWantsKey](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowIsVisible

```
ASBool AVWindowIsVisible (AVWindow win);
```

| | |
|------------------------|--|
| Description | Tests whether or not a window is being displayed on the screen. |
| Parameters | <p><i>win</i></p> <p>The window whose visibility is tested.</p> |
| Return Value | <i>true</i> if the window is being displayed on the screen (even if it is currently obscured by another window), <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVWindowBecomeKey AVWindowShow |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowMaximize

```
void AVWindowMaximize (AVWindow win, ASBool maximize);
```

| | |
|----------------------|---|
| Description | Maximizes the specified window. In Mac OS, this corresponds to calling the <i>ZoomWindow</i> Toolbox function. |
| Parameters | <p><i>win</i></p> <p>The window to maximize.</p> <p><i>maximize</i></p> <p><i>true</i> if the window is maximized, <i>false</i> if it is returned to its non-maximized state.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowNew

[AVWindow](#) AVWindowNew ([AVWindowLayer](#) layer, ASUns32 flags, [AVWindowHandler](#) handler, [ASExtension](#) owner);

| | |
|------------------------|---|
| Description | <p>Creates a new window and registers it with the Acrobat viewer.</p> <p>Because Windows and UNIX use the platform's native window handling instead of the Acrobat viewer's AVWindowHandler mechanism (that is, the <i>AVWindowHandler's</i> callbacks are never called on those platforms), there is no advantage to using <i>AVWindowNew</i>. Plug-in developers on those platforms should use AVWindowNewFromPlatformThing instead of this method.</p> |
| Parameters | <p><i>layer</i></p> <p>The layer in which the window resides. In Mac OS, must be one of the AVWindowLayer constants. In Windows, all <i>AVWindows</i> are of type <i>AVWLfloating</i>, and <i>layer</i> is ignored.</p> <p><i>flags</i></p> <p>An OR of the values listed in AVWindow Flags.</p> <p><i>handler</i></p> <p>A structure containing the window handler's callback functions. Pass <i>NULL</i> in Windows and UNIX, because the Window handler's callbacks are unused on those platforms.</p> <p><i>owner</i></p> <p>The gExtensionID extension registering the window.</p> |
| Return Value | The newly-created window. |
| Exceptions | genErrNoMemory |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVWindowNewFromPlatformThing |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowNewFromPlatformThing

```
AVWindow AVWindowNewFromPlatformThing (AVWindowLayer layer,  
ASUns32 flags, AVWindowHandler handler, ASExtension owner,  
void* platformThing);
```

Description

Creates a new window from a platform-specific structure and registers it with the Acrobat viewer.

If a user creates an *HWND* or a *WindowPtr* and does not register it with the viewer via *AVWindowNewFromPlatformThing*, it should not be allowed to persist across event loops (that is, in Mac OS, it should be system-modal).

Windows and UNIX use the platform's native window handling instead of the Acrobat viewer's [AVWindowHandler](#) mechanism (that is, the *AVWindowHandler*'s callbacks are *never* called on those platforms). Plug-in developers on those platforms should use *AVWindowNewFromPlatformThing* instead of [AVWindowNew](#), since they have to create the window using platform-specific code anyway.

Parameters

layer

One of the [AVWindowLayer](#) constants, specifying the layer in which the window is located. For Mac OS, see [AVWindowNew](#). *layer* is ignored in Windows, because the equivalent information was specified when the platform thing was created.

flags

For Mac OS, an OR of the [AVWindow Flags](#). It is the responsibility of the Macintosh plug-in developer to ensure that *flags* matches any attributes of the "platform-thing" window; the Acrobat viewer cannot determine *flags* from the window itself.

flags is ignored in Windows and UNIX, because the equivalent information was specified when the platform thing was created.

handler

A structure containing the window handler's callback functions.

Pass *NULL* in Windows and UNIX, because the Window handler's callbacks are unused on those platforms.

owner

The [gExtensionID](#) extension registering the window.

platformThing

A platform-specific object (*WindowPtr* in Mac OS, an *HWND* in Windows, and a *Widget* in UNIX) that will be used for this window.

Return Value The newly-created window.

Exceptions [genErrNoMemory](#)

Notifications None

Header File *AVCalls.h*

Related Methods [AVWindowNew](#)
[AVWindowGetPlatformThing](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowResignKey

```
void AVWindowResignKey (AVWindow win);
```

Description Use this method when you want *win* to resign its key window status. Another window might pick up key window status as a result. You must first call [AVWindowSetWantsKey](#) with a value of *false* for the *wantsKey* parameter or your window may immediately become the key window again.

UNIX users: This method is a no-op.

Parameters *win*
The window resigning key window status.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVWindowBecomeKey](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowSetFrame

```
void AVWindowSetFrame (AVWindow win, const AVRect* rect);
```

| | |
|------------------------|--|
| Description | Sets the window's frame, which specifies its size and location on the screen. |
| Parameters | <p><i>win</i></p> <p>The window whose frame is set.</p> <p><i>rect</i></p> <p>Pointer to a rectangle specifying the window's frame rectangle, specified in global screen coordinates.</p> <p>In Mac OS, the frame includes only the window's content region.</p> <p>In Windows, it includes the entire window (content region, title bar, and so forth).</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVWindowGetFrame AVWindowGetInterior |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowSetOwnerData

```
void AVWindowSetOwnerData (AVWindow win, void* newData);
```

Description Sets a window's owner data. The owner data is private data for the use of the window's creator. For example, if a plug-in uses its own class library, it might use the owner data field to store a pointer to the object owning the *AVWindow*.

Parameters

win
The window whose owner data is set.

newData
Pointer to the new owner data for *win*.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVWindowGetOwnerData](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowSetTitle

```
void AVWindowSetTitle (AVWindow win, const char* newTitle);
```

Description

Sets the title to display in the specified window's title bar. This method cannot be used to set the title of the window in which the Acrobat viewer normally opens a PDF file; it can only be used to set the title of an *AVWindow* created by a plug-in. To set the title of a window in which the Acrobat viewer opens a PDF file, you must replace [AVDocOpenFromFileWithParams](#) and pass the window title in *tempTitle*.

Parameters

win

The window whose title is set.

newTitle

The title to set for *win*.

Return Value

None

Exceptions

[genErrNoMemory](#)

Notifications

None

Header File

AVCalls.h

Related Methods

[AVWindowGetTitle](#)

Availability

Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowShow

```
void AVWindowShow (AVWindow win);
```

| | |
|------------------------|---|
| Description | Shows the specified window. |
| Parameters | <p><i>win</i></p> <p>The window to show.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | AVWindowHide AVWindowGetPlatformThing |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

AVWindowSetWantsKey

```
void AVWindowSetWantsKey (AVWindow win, ASBool wantsKey);
```

Description

Sets or clears a flag indicating that *win* wants to become the key window. This corresponds to the `AVWIN_WANTSKEY` flag that can be specified when the window is created using [AVWindowNew](#).

Once this flag is set, the Acrobat viewer may, at any time, make this window the key window. This occurs, for example, if the current key window closes and this window moves to the front. If *WantsKey* is *false*, this window will not become the key window. If the window is already the key window, however, simply invoking `AVWindowSetWantsKey` with *wantsKey* set to *false* will not cause it to resign key window status; to do that, you must call [AVWindowResignKey](#).

Parameters

win

The window that wants to become the key window.

wantsKey

If *true*, *win* is willing to become the key window, *false* otherwise.

Return Value

None

Exceptions

None

Notifications

None

Header File

`AVCalls.h`

Related Methods

[AVWindowNewFromPlatformThing](#)

Availability

Plug-ins: Available if `PI_ACROVIEW_VERSION` (in `PIRequir.h`) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

AVWindowUserClose

```
ASBool AVWindowUserClose (AVWindow win, ASBool quitting);
```

Description Simulates a user's click on a window's box. This calls the [AVWindowWillCloseProc](#) of *win*'s [AVWindowHandler](#).

Parameters *win*
The window to close.

quitting
If *true*, assume the application is trying to quit. If *false*, assume the user clicked on the window's close box.

Return Value *true* if the window was closed, *false* if the window handler's [AVWindowWillCloseProc](#) aborted the close by returning *false*.

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVWindowDestroy](#)

Availability Plug-ins: Available if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

Cos Layer

CosArray

CosArrayGet

[CosObj](#) CosArrayGet ([CosObj](#) array, ASInt32 index);

| | |
|------------------------|---|
| Description | Gets the specified element from an array. |
| Parameters | <p><i>array</i></p> <p>The array from which an element is obtained.</p> <p><i>index</i></p> <p>The array element to obtain. The first element in an array has an index of zero.</p> |
| Return Value | The Cos object occupying the <i>index</i> th element of <i>array</i> . Returns a <i>NULL</i> Cos object if <i>index</i> is outside the array bounds. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosArrayLength CosArrayPut CosArrayInsert |
| Example | <pre>ASInt32 i; CosObj newObj; for(i = 0 ;i <=CosArrayLength(cos)-1;i++) { newObj = CosArrayGet(cosArray, i); /*do something with newobj */ ... }</pre> |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

CosArrayInsert

```
void CosArrayInsert (CosObj array, ASInt32 pos, CosObj obj);
```

| | |
|------------------------|---|
| Description | Inserts an object into an array. |
| Parameters | <p><i>array</i></p> <p>The array into which the object is inserted.</p> <p><i>pos</i></p> <p>The location in the array to insert the object. The object is inserted before the specified location. The first element in an array has a <i>pos</i> of zero. If $pos \geq \text{CosArrayLength}(array)$, appends <i>obj</i> to <i>array</i> (increasing the array length by 1).</p> <p><i>obj</i></p> <p>The object to insert.</p> |
| Return Value | None |
| Exceptions | Raises an exception if object to insert is a direct object that is already contained in another object or if object to insert belongs to another document. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosArrayLength CosArrayRemove CosArrayGet |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosArrayLength

```
ASInt32 CosArrayLength (CosObj array);
```

| | |
|------------------------|--|
| Description | Gets the number of elements in <i>array</i> . |
| Parameters | <p><i>array</i></p> <p>The array for which the number of elements is determined.</p> |
| Return Value | The number of elements in <i>array</i> . |
| Exceptions | File access, memory, object type. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | Numerous |
| Example | <pre>if(CosArrayLength(cosArray) > MAXLEN) AVAlertNote("Array too long");</pre> |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosArrayPut

```
void CosArrayPut (CosObj array, ASInt32 index, CosObj obj);
```

Description

Puts the specified object into the specified location in an array. The array is extended as much as necessary and *NULL* objects are stored in empty slots. Sets the *PDDocNeedsSave* flag (see [PDDocSetFlags](#)) flag of *array*'s CosDoc if array is indirect or is a direct non-scalar object with an indirect composite object at the root of its container chain.

Parameters

array

The array in which *obj* is stored.

index

The location in *array* to store *obj*. The first element of an array has an index of zero.

obj

The Cos object to insert into *array*.

Return Value

None

Exceptions

Raises an exception if object to insert is a direct object that is already contained in another object or if object to insert belongs to another document.

Notifications

None

Header File

CosCalls.h

Related Methods

[CosArrayLength](#)
[CosArrayGet](#)
[CosArrayInsert](#)

Availability

Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

CosArrayRemove

```
void CosArrayRemove (CosObj array, CosObj obj);
```

Description

Finds the first element, if any, equal to the specified object and removes it from the array. [CosObjEqual](#) is used to determine whether or not an array element is equal to the specified object.

The array is compressed after removing the element. The compression is accomplished by moving each element following the deleted element to the slot with the next smaller index and decrementing the *array*'s length by 1.

Because the array is automatically compressed when an element is removed, it is not safe to call [CosArrayRemove](#) within your callback procedure for [CosObjEnum](#).

Parameters

array

The array from which *obj* is removed.

obj

The object to remove.

Return Value

None

Exceptions

File access, memory, object type.

Notifications

None

Header File

CosCalls.h

Related Methods

[CosArrayInsert](#)

Availability

Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----------|---------|
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosArrayRemoveNth

```
void CosArrayRemoveNth (CosObj array, ASInt32 pos);
```

Description Checks whether the position is within the array bounds and then removes it from the array and moves each subsequent element to the slot with the next smaller index and decrements the array's length by 1. Sets the "dirty" flag of array's *CosDoc*.

Parameters

array
The *CosArray* to remove the member from.

pos
The index for the array member to remove. Array indexes start at 0.

Return Value None

Exceptions None

Notifications None

Header File *CosCalls.h*

Related Methods [CosObjEqual](#)

Availability Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

CosNewArray

```
CosObj CosNewArray (CosDoc dP, ASBool indirect,
    ASInt32 nElements);
```

| | |
|------------------------|---|
| Description | Creates and returns a new array Cos object. |
| Parameters | <p><i>dP</i></p> <p>The document in which the array is used.</p> <p><i>indirect</i></p> <p>If <i>true</i>, creates the array as an indirect Cos object, and sets <i>dP</i>'s <i>PDDocNeedsSave</i> flag (see PDDocSetFlags).</p> <p>If <i>false</i>, creates the array as a direct object.</p> <p><i>nElements</i></p> <p>The number of elements that will be in the array. <i>nElements</i> is only a hint; Cos arrays grow dynamically as needed.</p> |
| Return Value | The newly-created array Cos object. |
| Exceptions | Raises an exception if storage is exhausted or file access fails. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosObjDestroy CosArrayGet CosArrayInsert CosArrayLength CosArrayPut CosArrayRemove |
| Example | <pre>CosObj cosArray = CosNewArray(NULL, false, 10);</pre> |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosBoolean

CosBooleanValue

```
ASBool CosBooleanValue (CosObj obj);
```

| | |
|------------------------|--|
| Description | Gets the value of the specified boolean object. |
| Parameters | <p><i>obj</i></p> <p>The boolean Cos object whose value is obtained.</p> |
| Return Value | The value of <i>obj</i> . |
| Exceptions | Raises an exception if <i>obj</i> has the wrong Cos type. Also raises exceptions if storage is exhausted or file access fails. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosNewBoolean |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosNewBoolean

```
CosObj CosNewBoolean (CosDoc dP, ASBool indirect,  
ASBool value);
```

| | |
|------------------------|--|
| Description | Creates a new boolean object associated with the specified document and having the specified value. |
| Parameters | <i>dP</i> The document in which the boolean is used. <i>indirect</i> If <i>true</i> , creates the boolean object as an indirect object, and sets the document <i>dP</i> 's <i>PDDocNeedsSave</i> flag (see PDDocFlags). If <i>false</i> , creates the boolean as a direct object. <i>value</i> The value the new boolean will have. |
| Return Value | A Cos boolean object. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosBooleanValue CosObjDestroy |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosDict

CosDictGet

```
CosObj CosDictGet (CosObj dict, ASAtom key);
```

Description

Gets the value of the specified key in the specified dictionary. Use [CosObjEnum](#) to list all keys in a dictionary.

This method can also be called with a stream object instead of a dictionary object. In that case, this method gets the value of the specified key from the stream's attributes dictionary.

Parameters

dict

The dictionary or stream from which a value is obtained.

key

The key whose value is obtained. A string can be converted to an *ASAtom* using [ASAtomFromString](#).

See the [Portable Document Format Reference Manual](#) to obtain the names of keys in dictionary objects that are part of standard PDF, such as annotations or page objects.

Note: Although a dictionary key is a name object in a PDF file and names begin with the slash character "/", the key parameter should not start with "/"—unless the key actually begins with "/".

Return Value

The object associated with the specified key. Returns a *NULL* Cos object if *key* is not present unless the desired *key* is *AcroForm*. In that case, a new *AcroForm* dictionary is created and returned.

Exceptions

None

Notifications

None

Header File

CosCalls.h

Related Methods

[CosDictPut](#)
[CosDictKnown](#)
[CosStreamDict](#)

Example

```
CosObj theDict, intObj;

intObj = CosDictGet(theDict,
    ASAtomFromString ("Key1"));
```

Availability

Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosDictKnown

```
ASBool CosDictKnown (CosObj dict, ASAtom key);
```

Description

Tests whether or not a specific key is found in the specified dictionary. Calling this method is equivalent to checking if the value returned from [CosDictGet](#) is a *NULL* Cos object.

Use [CosObjEnum](#) to obtain a list of all keys in a dictionary.

Parameters

dict

The dictionary in which to look for *key*.

key

The key to find. A string can be converted to an *ASAtom* using [ASAtomFromString](#).

See the [Portable Document Format Reference Manual](#) to obtain the names of keys in dictionary objects that are part of standard PDF, such as annotations or page objects.

Note: Although a dictionary key is a name object in a PDF file and names begin with the slash character “/”, the key parameter should not start with “/”—unless the key actually begins with “/”.

Return Value

true if key is found in *dict*, *false* otherwise.

Exceptions

None

Notifications

None

Header File

CosCalls.h

Related Methods

[CosDictGet](#)
[CosDictPut](#)

Example

```
CosObj myDict;

if (CosDictKnown(myDict, ASAtomFromString
    ("Optional") == true)
    {
        /* The key is in the dictionary. */
        ...
    }
else
    {
        /* That key is not in the dictionary. */
        ...
    }
}
```

Availability

Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosDictPut

```
void CosDictPut (CosObj dict, ASAtom key, CosObj val);
```

Description

Sets the value of a dictionary key, adding the key to the dictionary if it is not already present. Sets the *PDDocNeedsSave* flag (see [PDDocSetFlags](#)) of *dict*'s *CosDoc* if *dict* is indirect or is a direct non-scalar object with an indirect composite object at the root of its container chain.

This method can also be used with a stream object. In that case, the key–value pair is added to the stream's attributes dictionary.

Parameters

dict

The dictionary or stream in which a value is set.

key

The key whose value is set. A string can be converted to an *ASAtom* using [ASAtomFromString](#). *key* must not contain any white space characters (for example, spaces or tabs).

See the [Portable Document Format Reference Manual](#) to obtain the names of keys in dictionary objects that are part of standard PDF, such as annotations or page objects.

Note: Although a dictionary key is a name object in a PDF file and names begin with the slash character “/”, the key parameter should not start with “/”—unless the key actually begins with “/”.

val

The value to set.

Return Value

None

Exceptions

Raises an exception if object to insert is a direct object that is already contained in another object or if object to insert belongs to another document.

Notifications

None

Header File

CosCalls.h

Related Methods [CosDictGet](#)
[CosDictKnown](#)
[PDDocSetFlags](#)
[CosStreamDict](#)

Example `CosObj dict, intObj;`

```
CosDictPut(dict, ASAtomFromString("Key1"),  
           intObj);
```

Availability Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*)
is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosDictRemove

```
void CosDictRemove (CosObj dict, ASAtom key);
```

Description

Removes a key–value pair from a dictionary. Sets the *PDDocNeedsSave* flag (see [PDDocSetFlags](#)) of *dict*'s *CosDoc* if the dictionary is indirect or has an indirect composite object at the root of its container chain.

Because of the way in which key–value pairs are represented in memory, it is not safe to call [CosDictRemove](#) within your callback procedure for [CosObjEnum](#).

Parameters

dict

The dictionary from which the key–value pair is removed.

key

The key to remove. A string can be converted to an *ASAtom* using [ASAtomFromString](#).

See the [Portable Document Format Reference Manual](#) to obtain the names of keys in dictionary objects that are part of standard PDF, such as annotations or page objects.

Note: Although a dictionary key is a name object in a PDF file and names begin with the slash character “/”, the key parameter should not start with “/”—unless the key actually begins with “/”.

Return Value

None

Exceptions

None

Notifications

None

Header File

CosCalls.h

Related Methods

[CosDictPut](#)

Example

```
CosObj dict;
```

```
CosDictRemove(dict,
ASAtomFromString("MyKey"));
```

Availability

Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosNewDict

```
CosObj CosNewDict (CosDoc dP, ASBool indirect,
    ASInt32 nEntries);
```

| | |
|------------------------|--|
| Description | <p>Creates a new dictionary.</p> <p>See the Portable Document Format Reference Manual for information on dictionary objects that are part of standard PDF, such as annotations or page objects.</p> |
| Parameters | <p><i>dP</i></p> <p>The document in which the dictionary is used.</p> <p><i>indirect</i></p> <p>If <i>true</i>, creates the dictionary as an indirect Cos object, and sets <i>dP</i>'s <i>PDDocNeedsSave</i> flag (see PDDocFlags).</p> <p>If <i>false</i>, creates the dictionary as a direct object.</p> <p><i>nEntries</i></p> <p>Number of entries in the dictionary. This value is only a hint—Cos dictionaries grow dynamically as needed.</p> |
| Return Value | The newly-created dictionary Cos object. |
| Exceptions | Raises an exception if storage is exhausted or file access fails. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosObjDestroy CosDictGet CosDictKnown CosDictPut CosDictRemove |
| Example | <pre>PDDoc d; CosDoc cd; CosObj dict; cd = PDDocGetCosDoc(d); dict = CosNewDict(cd, false, 2);</pre> |

Availability

Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosDoc

CosDocClose

```
void CosDocClose ( CosDoc cosDoc );
```

| | |
|------------------------|--|
| Description | Closes a Cos document. You should only call this method with a document obtained via CosDocOpenWithParams to release resources used by the Cos document. |
| Parameters | <i>cosDoc</i> The document to close. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosDocOpenWithParams |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosDocCreate

[CosDoc](#) CosDocCreate (ASUns32 createFlags);

| | |
|------------------------|--|
| Description | Creates an empty Cos document. |
| Parameters | <p><i>createFlags</i></p> <p>An OR of the values listed in CosDocCreate Flags specifying how to create the document.</p> |
| Return Value | An empty Cos document. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | CosDocSaveToFile |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosDocEnumEOFs

```
ASBool CosDocEnumEOFs (CosDoc cosDoc, CosDocEnumEOFsProc proc,
    void* clientData);
```

| | |
|------------------------|---|
| Description | Calls CosDocEnumEOFsProc for each EOF in a given <i>CosDoc</i> . |
| Parameters | <p><i>cosDoc</i></p> <p>The <i>CosDoc</i> in which the EOFs are enumerated.</p> <p><i>proc</i></p> <p>The CosDocEnumEOFsProc to call for each EOF.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>proc</i> each time it is called.</p> |
| Return Value | <i>true</i> if all of the calls to <i>proc</i> return <i>true</i> . <i>false</i> as soon as a call to <i>proc</i> returns <i>false</i> . |
| Exceptions | Raises an exception if storage is exhausted or file access fails. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosDocCreate CosDocEnumIndirect CosDocSaveToFile CosDocSaveWithParams |
| Example | <pre>static ACCB1 ASBool ACCB2 myCosObjEnumDictProc(CosDoc cosDoc, ASInt32 fileOffset, void* clientData); { ...process this portion of the CosDoc... return true; } ... PDDoc myPdDoc; CosDocEnumEOFsProc myEnumCB = ASCallbackCreateProto(CosDocEnumEOFsProc, &myCosObjEnumDictProc); CosDocEnumEOFs(PDDocGetCosDoc(myPdDoc), myEnumCB, &clientData);</pre> |

Availability

Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|---------|
| Adobe PDF Library | — | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | — | M, W |
| Reader | — | M, W, U |

CosDocEnumIndirect

```
ASBool CosDocEnumIndirect (CosDoc dP, CosObjEnumProc proc,
    void* clientData);
```

Description

Enumerates all the indirect objects of a given *CosDoc*.

The objects are enumerated in no particular order. Successive enumerations of the same Cos document are not guaranteed to enumerate objects in the same order.

CosDocEnumIndirect does not enumerate "missing" objects. For example, a document might include a reference "23 0 R" without a definition "23 0 obj ... endobj". This is equivalent to defining that object to be *NULL*, but the enumeration procedure is not called with the *NULL* value in this case. If an indirect object is explicitly defined to be *NULL*, for example, "47 0 obj *NULL* endobj", the enumeration procedure is called with the *NULL* value.

Parameters

dP

The *CosDoc* whose indirect objects are enumerated.

proc

User-supplied callback to call for each indirect object in *dP*. Enumeration ends when *proc* returns *false* or all indirect objects have been enumerated.

The *value* parameter returned in *proc* is always the *NULL* Cos object.

clientData

Pointer to user-supplied data to pass to *proc* each time it is called.

Return Value

true if all of the calls to *proc* returned *true*. *false* as soon as a call to *proc* returns *false*.

Exceptions

Raises an exception if storage is exhausted or file access fails.

Also re-raises any exception that *proc* raises.

Notifications

None

Header File

CosCalls.h

Related Methods

[CosObjEnum](#)

Example

```
static ACCB1 ASBool ACCB2
myCosObjEnumProc(CosObj obj,
CosObj value, void* clientData)
{
...process the indirect object...
return true;
}
...
PDDoc myPdDoc;

CosObjEnumProc myEnumCB =
ASCallbackCreateProto(CosObjEnumProc,
&myCosObjEnumProc);
CosDocEnumIndirect(
PDDocGetCosDoc(myPdDoc), myEnumCB,
&clientData);
```

Availability

Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

CosDocGetInfoDict

```
CosObj CosDocGetInfoDict (CosDoc dP);
```

| | |
|------------------------|---|
| Description | Gets the specified document's Info dictionary. In general, access the document's Info dictionary using PDDocGetInfo and PDDocSetInfo wherever possible. |
| Parameters | <i>dP</i> The document whose Info dictionary is obtained. |
| Return Value | The document's Info dictionary Cos object. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosDocGetObjByID |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosDocGetObjByID

[CosObj](#) CosDocGetObjByID ([CosDoc](#) dP, ASUns32 objNum);

| | |
|------------------------|---|
| Description | Gets the indirect <i>CosObj</i> with the latest generation number matching the specified local master index (ID). For indirect objects, the local master index is the same as the indirect object index that appears in the PDF file. Returns <i>CosNewNull</i> if there is no object with this ID. |
| Parameters | <p><i>dP</i></p> <p>The <i>CosDoc</i> to search for the matching Cos Object.</p> <p><i>objNum</i></p> <p>The local master index for the indirect Cos Object to return.</p> |
| Return Value | The <i>CosObj</i> matching the specified local master index or <i>CosNewNull</i> if there is no object with this ID. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosObjGetID |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

CosDocGetRoot

```
CosObj CosDocGetRoot ( CosDoc dP );
```

| | |
|------------------------|---|
| Description | Gets the Catalog (the root object) for the specified document. See Section 6.2 in the Portable Document Format Reference Manual for a description of the Catalog. |
| Parameters | <i>dP</i> The document whose Catalog is obtained. |
| Return Value | The document's Catalog dictionary Cos object. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosDocGetInfoDict |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosDocOpenWithParams

[CosDoc](#) CosDocOpenWithParams ([CosDocOpenParams](#) params);

Description

Opens a Cos document. The document does not need to be a PDF document. The client specifies a fileSys and pathName from which to open the document. The client may also specify a header string other than "%PDF-". For example, a client might want to open a private file type, for example, "%FDF-".

If the *doRepair* flag is set in the open flags, a minimal document can be opened. A minimal document contains the header string and a trailer dictionary. It may contain indirect objects before the trailer dictionary, and the trailer dictionary can refer to those objects. For example:

```
%FDF-1.0
1 0 obj
<<
/AcroForm <</This /Is /An /Example>>
>>
trailer
<<
/Root 1 0 R
>>
```

Parameters

params

Specifies how to open the document.

Return Value

A Cos document.

Exceptions

Various

Notifications

None

Header File

CosCalls.h

Related Methods

[CosDocClose](#)

Availability

Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosDocSaveToFile

```
void CosDocSaveToFile (CosDoc cosDoc, ASFile asFile,
    ASUns32 saveFlags, CosDocSaveParams saveParams);
```

| | |
|------------------------|--|
| Description | Saves a Cos document to a file handle. |
| Parameters | <p><i>cosDoc</i></p> <p>Document to save.</p> <p><i>asFile</i></p> <p>File to which the document is written; must be open in write mode. This file is not necessarily positionable.</p> <p><i>saveFlags</i></p> <p>An OR of the values listed in CosDocSave Flags specifying how to save the document.</p> <p><i>saveParams</i></p> <p>Optional parameters for use when saving a document, as described in CosDocSaveParams.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | CosDocCreate CosDocSaveWithParams |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosDocSaveWithParams

```
void CosDocSaveWithParams (CosDoc cosDoc, ASFile asFile, ASUns32
    saveFlags, CosDocSaveParams saveParams);
```

Description Saves a Cos document, optionally to a new file handle.

Parameters

cosDoc

The *CosDoc* for the document to save.

asFile

(Optional) If saving to the same file, do not pass in an *ASFile*. If saving to a different file, specify the file to which the document is written; it must be open in write mode.

If *NULL*, this method saves to the *ASFile* originally associated with the *CosDoc*.

saveFlags

A bit field composed of the [CosDocSave Flags](#). (See *CosExpT.h*.)

saveParams

Optional [CosDocSaveParams](#) parameters for use when saving the *CosDoc* document. (See *CosExpT.h*.)

Return Value None

Exceptions

[cosErrAfterSave](#)
[cosErrNeedFullSave](#)
[genErrBadParm](#)

Notifications None

Header File *CosCalls.h*

Related Methods [CosDocCreate](#)
[CosDocSaveToFile](#)

Availability Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | — | M, W |
| Reader | — | M, W, U |

CosDocSetDirty

```
void CosDocSetDirty (CosDoc cosDoc, ASBool isDirty);
```

| | |
|------------------------|--|
| Description | Sets a Cos document's dirty flag to a given boolean value. |
| Parameters | <p><i>cosDoc</i></p> <p>The Cos document whose dirty flag is set.</p> <p><i>isDirty</i></p> <p><i>true</i> if dirty, <i>false</i> otherwise.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosFixed

CosFixedValue

```
ASFixed CosFixedValue (CosObj obj);
```

| | |
|------------------------|--|
| Description | Gets the value of the specified fixed number object. It is legal to call CosIntegerValue on a fixed number object and CosFixedValue on an integer object. In fact, a fixed number object whose value is an integer is stored as an integer. If the document is saved and reopened, the object's type will be integer. |
| Parameters | <i>obj</i> The object whose value is obtained. |
| Return Value | The value of <i>obj</i> . |
| Exceptions | Raises an exception if the given object has the wrong Cos type. Also raises exceptions if storage is exhausted or file access fails. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosIntegerValue CosNewFixed |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosNewFixed

```
CosObj CosNewFixed (CosDoc dP, ASBool indirect, ASFixed value);
```

Description Creates a new fixed number object, associated with the specified document and having the specified *value*.

Parameters

dP

The document in which the fixed number is used.

indirect

If *true*, creates the fixed number object as an indirect object, and sets the document *dP*'s *PDDocNeedsSave* flag (see [PDDocFlags](#)).

If *false*, creates the fixed number as a direct object.

value

The value the new fixed number will have.

A fixed number object whose value is an integer is stored as an integer. If the document is saved and reopened, the object's type will be integer.

Return Value A fixed Cos object.

Exceptions None

Notifications None

Header File *CosCalls.h*

Related Methods [CosObjDestroy](#)
[CosFixedValue](#)
[CosNewInteger](#)

Availability Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosInteger

CosIntegerValue

```
ASInt32 CosIntegerValue (CosObj obj);
```

| | |
|------------------------|---|
| Description | Gets the integer value of the specified number object. It is legal to call CosIntegerValue on a fixed number object and CosFixedValue on an integer object. In fact, a fixed number object whose value is an integer will be stored as an integer. If the document is saved and reopened, its type will be integer. |
| Parameters | <i>obj</i> The integer object whose value is obtained. |
| Return Value | The value of <i>obj</i> . |
| Exceptions | Raises an exception if the given object has the wrong Cos type. Also raises exceptions if storage is exhausted or file access fails. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosFixedValue CosNewFixed CosNewInteger |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosNewInteger

```
CosObj CosNewInteger (CosDoc dP, ASBool indirect,  
ASInt32 value);
```

| | |
|------------------------|---|
| Description | Creates a new integer object associated with the specified document and having the specified value. |
| Parameters | <p><i>dP</i></p> <p>The document in which the integer is used.</p> <p><i>indirect</i></p> <p>If <i>true</i>, creates the integer object as an indirect object, and sets the document <i>dP</i>'s <i>PDDocNeedsSave</i> flag (see PDDocFlags).</p> <p>If <i>false</i>, creates the integer as a direct object.</p> <p><i>value</i></p> <p>The value the new integer will have.</p> |
| Return Value | An integer Cos object. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosObjDestroy CosIntegerValue CosNewFixed |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosName

CosNameValue

[ASAtom](#) CosNameValue ([CosObj](#) obj);

| | |
|------------------------|--|
| Description | Gets the value of the specified name object. |
| Parameters | <p><i>obj</i></p> <p>The object whose value is obtained.</p> |
| Return Value | The <i>ASAtom</i> corresponding to the specified name object. <i>ASAtom</i> can be converted to a string using ASAtomGetCount . |
| Exceptions | Raises an exception if <i>obj</i> has the wrong type, or it is an invalid object. Also raises exceptions if storage is exhausted or file access fails. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | ASAtomGetCount CosNewName |
| Example | <pre>ASAtom nameAtom = CosNameValue(cosObj); if (nameAtom == ASAtomFromString("Roger")) { ... }</pre> |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosNewName

```
CosObj CosNewName (CosDoc dP, ASBool indirect, ASAtom name);
```

| | |
|------------------------|--|
| Description | Creates a new name object associated with the specified document and having the specified value. |
| Parameters | <p><i>dP</i></p> <p>The document in which the new name is used.</p> <p><i>indirect</i></p> <p>If <i>true</i>, creates the name as an indirect object, and sets the document's <i>PDDocNeedsSave</i> flag (see PDDocFlags) flag.</p> <p>If <i>false</i>, creates the name as a direct object.</p> <p><i>name</i></p> <p>The <i>ASAtom</i> corresponding to the name to create. <i>name</i> must not contain any white space characters (for example, spaces or tabs). A C string can be converted to an <i>ASAtom</i> using ASAtomFromString.</p> |
| Return Value | The newly-created name Cos object. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosObjDestroy CosNameValue |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosNull

CosNewNull

CosObj CosNewNull (void);

| | |
|------------------------|---|
| Description | Gets a <i>NULL</i> Cos object. For efficiency, the Acrobat viewer has only one <i>NULL</i> object, which is shared by all methods that request a <i>NULL</i> object. For this reason, the <i>NULL</i> object cannot be destroyed. |
| Parameters | None |
| Return Value | A <i>NULL</i> Cos object. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | None |
| Example | <pre>dObj = CosDictGet(cosDict, ASAtomFromString("D")); if (CosObjEqual(dObj, CosNewNull()) AVAlertNote("Expected non-NULL value");</pre> |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosObj

CosObjCopy

```
CosObj CosObjCopy (CosObj srcObj, CosDoc destDoc, ASBool  
copyIndirect);
```

| | |
|------------------------|--|
| Description | Copies a <i>CosObj</i> from one document to another (or the same document). |
| Parameters | <p><i>srcObj</i></p> <p>The <i>CosObj</i> to copy.</p> <p><i>destDoc</i></p> <p>The <i>CosDoc</i> for the document into which the <i>CosObj</i> is copied.</p> <p><i>copyIndirect</i></p> <p><i>true</i> if all indirectly-referenced objects from <i>srcObj</i> are copied to <i>destDoc</i>, <i>false</i> otherwise.</p> |
| Return Value | The <i>CosObj</i> which has been copied to the destination document. |
| Exceptions | Raises an exception if storage is exhausted or file access fails. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosObjEqual |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

CosObjDestroy

```
void CosObjDestroy (CosObj obj);
```

| | |
|------------------------|--|
| Description | Destroys a Cos object. This method does nothing if <i>obj</i> is a <i>NULL</i> Cos object or a direct scalar Cos object. |
| Parameters | <i>obj</i> The object to destroy. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosNewBoolean CosNewInteger CosNewFixed CosNewString CosNewName CosArrayRemoveNth CosNewDict CosNewStream |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosObjEnum

```
ASBool CosObjEnum (CosObj obj, CosObjEnumProc proc,  
void* clientData);
```

| | |
|------------------------|--|
| Description | Enumerates the elements of a Cos object by calling a user-supplied procedure for each component of the object. See CosObjEnum Actions for a description of what this means for each Cos object type. |
| Parameters | <p><i>obj</i></p> <p>The object whose elements are enumerated.</p> <p><i>proc</i></p> <p>User-supplied callback to call for each element of <i>obj</i>. Enumeration ends if <i>proc</i> returns <i>false</i>.</p> <p><i>proc</i> must not call CosArrayRemove (if <i>obj</i> is an array) or CosDictRemove (if <i>obj</i> is a dictionary).</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>proc</i> each time it is called.</p> |
| Return Value | Returns the value that <i>proc</i> returned (that is, returns <i>true</i> if all the elements of the object were enumerated, <i>false</i> if enumeration was terminated at the request of <i>proc</i>). |
| Exceptions | Raises an exception if storage is exhausted or file access fails. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosArrayGet CosDictGet CosDocEnumIndirect |

Example

```
static ACCB1 ASBool ACCB2
    myCosObjEnumDictProc(CosObj key,
        CosObj value, void* clientData)
{
    ...process the key-value pair...
    return true;
}

ASCallback myEnumCB =
    ASCallbackCreateProto(CosObjEnumProc,
        &myCosObjEnumDictProc);
CosObjEnum(cosobjDict, myEnumCB,
    &clientData);
```

Availability

Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosObjEqual

```
ASBool CosObjEqual (CosObj obj1, CosObj obj2);
```

Description

Tests whether or not two Cos objects are equal.

Two indirect or non-scalar Cos objects are equal if they have the same generation number (see Section 5.4 in the [Portable Document Format Reference Manual](#)), the same type, and reference the same object.

Two direct scalar Cos objects are equal if they have the same value and the same type. The generation number of direct scalar objects is always zero.

Two *NULL* Cos objects are always equal.

Parameters

obj1, obj2

Objects to compare.

Return Value

true if *obj1* and *obj2* are equal, *false* otherwise.

Exceptions

Raises an exception if storage is exhausted or file access fails.

Notifications

None

Header File

CosCalls.h

Related Methods

None

Availability

Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosObjGetDoc

```
CosDoc CosObjGetDoc ( CosObj obj );
```

| | |
|------------------------|--|
| Description | Gets the <i>CosDoc</i> containing the specified object. This is defined <i>only</i> for indirect or non-scalar objects. |
| Parameters | <i>obj</i> The object whose <i>CosDoc</i> is obtained. |
| Return Value | The object's <i>CosDoc</i> . |
| Exceptions | Raises cosErrInvalidObj if the obj is not contained in a <i>CosDoc</i> , or if the object is a direct scalar object. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | PDDocGetCosDoc |
| Example | <code>CosDoc myCosDoc = CosObjGetDoc(cosObj);</code> |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosObjGetGeneration

```
ASUns16 CosObjGetGeneration (CosObj obj);
```

| | |
|------------------------|--|
| Description | Gets the generation number of an indirect Cos object. See Section 5.4 in the Portable Document Format Reference Manual for more information. |
| Parameters | <i>obj</i> The indirect <i>CosObj</i> for which the generation number is obtained. A <i>CosObj</i> can be determined to be indirect using CosObjsIndirect . |
| Return Value | The generation number of <i>cosObj</i> . |
| Exceptions | Raises cosErrInvalidObj if the object is not valid or is not indirect. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosObjsIndirect |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

CosObjGetID

```
ASUns32 CosObjGetID (CosObj obj);
```

| | |
|------------------------|--|
| Description | Gets the local master index for an indirect object. For indirect objects, the local master index is the same as the indirect object index that appears in the PDF file. |
| Parameters | <p><i>obj</i></p> <p>The indirect <i>CosObj</i> for which the ID is obtained. A <i>CosObj</i> can be determined to be indirect using CosObjIsIndirect.</p> |
| Return Value | The ID of <i>obj</i> . |
| Exceptions | Raises cosErrInvalidObj if the object is not valid or is not indirect. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosDocGetObjByID CosObjGetGeneration CosObjIsIndirect |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

CosObjGetType

```
CosType CosObjGetType ( CosObj obj );
```

| | |
|------------------------|--|
| Description | Gets an object's type. |
| Parameters | <p><i>obj</i></p> <p>The object whose type is obtained.</p> |
| Return Value | The object's type. Must be one of the Cos Object Types . |
| Exceptions | Raises an exception if storage is exhausted or file access fails. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosObjHash

```
ASUns32 CosObjHash (CosObj obj);
```

| | |
|------------------------|--|
| Description | Gets a 32-bit hash-code for the given <i>CosObj</i> . Two <i>CosObjects</i> with equal hash-codes are not necessarily equal, however. |
| Parameters | <i>obj</i> The <i>CosObj</i> for which to obtain a hash-code. |
| Return Value | 32-bit hash-code for the given <i>CosObj</i> , or <i>CosNewNull</i> if there is no object with this ID. |
| Exceptions | Raises an exception if storage is exhausted or file access fails. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosObjEqual |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

CosObjIsIndirect

```
ASBool CosObjIsIndirect (CosObj obj);
```

| | |
|------------------------|--|
| Description | Tests whether or not an object is indirect. |
| Parameters | <i>obj</i> The object to test. |
| Return Value | <i>true</i> if <i>obj</i> is indirect, <i>false</i> if <i>obj</i> is direct. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | None |
| Example | <pre>if(CosObjIsIndirect(cosObj)) { /* Process indirect objects only */ ... }</pre> |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosStream

CosNewStream

```
CosObj CosNewStream (CosDoc dP, ASBool indirect, ASStm stm,  
ASInt32 stmStartPos, ASBool stmDataIsDecoded,  
CosObj attributesDict, CosObj encodeParms,  
ASInt32 decodeLength);
```

Description

Creates a new Cos stream, using data from an existing *ASStm*. In the process of creating the Cos stream, the *ASStm*'s data is copied, so *stm* may be closed after *CosNewStream* returns.

You cannot call [CosStreamPos](#) on a stream created with *CosNewStream* until the file has been saved.

Parameters

dP

The Cos document in which the newly-created stream will be used.

indirect

Must always be *true*, specifying that the Cos stream is created as an indirect object. This also sets the document's *PDDocNeedsSave* flag (see [PDDocFlags](#)).

stm

The data to put into the stream. The user is responsible for closing *stm* after *CosNewStream* returns. *stm* can come from a file ([ASFileStmRdOpen](#)), from memory ([ASMemStmRdOpen](#)), or an arbitrary procedure ([ASProcStmRdOpen](#)).

stmStartPos

Specifies whether or not the stream is seekable. If the stream is seekable, also specifies the offset into *stm* from which data reading starts.

The sign of *stmStartPos* specifies whether or not the stream is seekable. A positive sign (or a value of zero) implies a seekable stream and a negative sign implies a non-seekable stream.

[ASFileStmRdOpen](#) and [ASMemStmRdOpen](#) always provide seekable streams, while [ASProcStmRdOpen](#) always provides a non-seekable stream. If you use [ASProcStmRdOpen](#) to create the *ASStm*, but the sign of *stmStartPos* indicates that the stream is seekable, an exception is raised.

If the stream is seekable, the absolute value of *stmStartPos* specifies the byte offset to seek to before reading data. Specifying a value of zero seeks to the beginning of the file, while a nonzero value allows you, for example, to skip header bytes in *stm*.

If the stream is *not* seekable, the magnitude of *stmStartPos* is ignored, and data is read starting at the current position.

stmDataIsDecoded

Determines whether or not the data in *stm* should be passed through the filters specified in *attributesDict* when copied to the Cos stream.

stmDataIsDecoded should be *false* if the data has been encoded by the filters specified by *attributesDict*. If the data has been decoded (or never encoded), *stmDataIsDecoded* should be *true*. If there are no filters, *stmDataIsDecoded* should be *true* if the stream length is specified by *decodeLength* and *false* if it is not.

If *stmDataIsDecoded* is *true*, the length of the decoded stream data should be specified in *decodeLength*.

In general, *stmDataIsDecoded* should be set to *true*. An exception would be if *stm* has been received from a fax modem and is already CCITT group 4 encoded.

attributesDict

A dictionary containing stream attributes, such as the length of the stream and a list of filters to apply to the data, as defined in Section 4.8 in the [Portable Document Format Reference Manual](#). The attributes dictionary must have been created as a *direct* (not indirect) object.

The value of the **Filter** entry of the attributes dictionary can be either the name of a single decode filter or an array of decode filter names. Specify multiple filters in the order they should be applied to decode the data.

Because of subtle interactions between *decodeLength* and the *Length* key in *attributesDict*, it is safest to let *CosNewStream* calculate the stream length. The easiest way to do this is to omit the *Length* key from *attributesDict*, or simply pass a *NULL* Cos object for *attributeDict* if you do not need it for other reasons.

If you decide to create a *Length* key in *attributesDict*, its value must be an indirect integer Cos object if the stream's length is not known. It may be either an indirect or a direct integer Cos object if you do know the stream's length (a direct integer is more efficient). *CosNewStream* uses a direct integer whenever possible if you let it create the *Length* key-value pair.

encodeParms

The parameters to be used by the filters when the data needs to be encoded before it is written to the file (that is, when *stmDataIsDecoded* is true).

If no filter encode parameters are needed, *encodeParms* should be a *NULL* Cos object. If filter encode parameters are used, follow the structure for the value of the *DecodeParms* stream attribute described in Table 4.2 in the [Portable Document Format Reference Manual](#).

Some filters use encode parameters that are *not* the same as the decode parameters. In this case, the decode parameters should be specified in *attributesDict* as the value of the **DecodeParms** key.

decodeLength

The amount of data in *stm* before it passed through any encoding filters. *decodeLength* is used only if *stmDataIsDecoded* is *true*.

If *decodeLength* = -1, data is read from *stm* until its *filbuf* returns 0 bytes, indicating *EOF*.

If *decodeLength* ≥ 0, at most *decodeLength* bytes are read from *stm*.

Return Value

The newly-created stream Cos object.

Exceptions

Raises [cosErrExpectedDict](#) if *attributesDict* is not a dictionary. Raises an exception if storage is exhausted. Raises an exception if *stmStartPos* ≥ 0 (requesting that *stm* seek to the specified position before reading) and *stm* is not seekable.

Raises [genErrGeneral](#) if *streamStartPos* is ≥ 0 (implying that it is seekable), but the stream is not really seekable. See the description of *stmStartPos* for more information about seekable and non-seekable streams.

Notifications

None

Header File

CosCalls.h

Related Methods

[CosObjDestroy](#)

Availability

Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosStreamDict

```
CosObj CosStreamDict (CosObj stream);
```

| | |
|------------------------|---|
| Description | Gets a stream's attributes dictionary. |
| Parameters | <p><i>stream</i></p> <p>The stream whose attributes dictionary is obtained.</p> |
| Return Value | The stream's attributes dictionary Cos object. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosStreamLength CosStreamPos CosDictGet CosDictPut |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosStreamLength

```
ASInt32 CosStreamLength (CosObj stream);
```

| | |
|------------------------|--|
| Description | Gets the length of a Cos stream. |
| Parameters | <p><i>stream</i></p> <p>The stream whose length is obtained.</p> |
| Return Value | The value of the <i>Length</i> key in the stream's attributes dictionary. |
| Exceptions | Raises an exception if the <i>Length</i> key is not found in the attributes dictionary or its value is not an integer. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosStreamDict CosStreamPos |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosStreamOpenStm

```
ASStm CosStreamOpenStm ( CosObj stream, CosStreamOpenMode mode );
```

Description Creates a new, non-seekable *ASStm* for reading data from a Cos stream. The data in the Cos stream may be filtered and encrypted. After opening the Cos stream, data can be read from it into memory using [ASStmRead](#). When reading is completed, close the stream using [ASStmClose](#).

Parameters

stream

The Cos stream for which an *ASStm* is opened.

mode

Must be one of the [CosStreamOpenMode](#) values.

Return Value The newly-opened *ASStm*.

Exceptions Raises [genErrGeneral](#) if in the call to [CosNewStream](#), *stmStartPos* ≥ 0 (that is, *stm* is requested to seek to the specified position before reading) and *stm* is not seekable.

Notifications None

Header File *CosCalls.h*

Related Methods [ASStmWrite](#)
[CosNewStream](#)

Availability Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosStreamPos

```
ASInt32 CosStreamPos (CosObj stream);
```

Description

Gets the byte offset of the start of a Cos stream's data in the PDF file (that is, the byte offset of the beginning of the line following the "stream" token). Use this method to obtain the file location of any private data in a stream that you need to read directly rather than letting it pass through the normal Cos mechanisms. For example, a QuickTime video embedded in a PDF file.

CosStreamPos is only valid when called on a stream that is already stored in a PDF document. If the stream was created using [CosNewStream](#), the new stream is stored in the document's temp file, and you *cannot* invoke *CosStreamPos* on it. After the file has been saved, you can use *CosStreamPos* on the stream.

Parameters

stream

The stream whose current position is obtained.

Return Value

The byte offset of the start of the Cos stream's data in the PDF file.

Exceptions

Raises [cosErrInvalidObj](#) if the stream object has not yet been saved to the PDF file. In other words, before you can call *CosStreamPos* on a newly-created stream, you must first save the PDF file.

Notifications

None

Header File

CosCalls.h

Related Methods

[CosStreamDict](#)
[CosStreamLength](#)

Availability

Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

CosString

CosNewString

```
CosObj CosNewString (CosDoc dP, ASBool indirect, char* str,
    ASInt32 nBytes);
```

| | |
|------------------------|---|
| Description | Creates and returns a new Cos string object. |
| Parameters | <p><i>dP</i></p> <p>The document in which the string is used.</p> <p><i>indirect</i></p> <p>If <i>true</i>, creates the string as an indirect object, and sets the document <i>dP</i>'s <i>PDDocNeedsSave</i> flag (see PDDocFlags).</p> <p>If <i>false</i>, creates the string as a direct object.</p> <p><i>str</i></p> <p>The value that the new string will have. It is <i>not</i> a C string, since Cos strings can contain <i>NULL</i> characters. The data in <i>str</i> is copied, that is, if <i>str</i> was dynamically allocated, it can be freed after this call.</p> <p><i>nBytes</i></p> <p>The length of <i>str</i>.</p> |
| Return Value | The newly-created string Cos object. |
| Exceptions | Raises an exception if storage is exhausted or file access fails. |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosStringValue CosObjDestroy |
| Availability | Plug-ins: Available if <i>PI_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosStringGetHexFlag

```
ASBool CosStringGetHexFlag (CosObj cosObj);
```

| | |
|------------------------|---|
| Description | Gets the hex flag of the <i>CosString</i> . The hex flag specifies whether the <i>CosString</i> should be written out as hex when writing the Cos Object to file. |
| Parameters | <p><i>cosObj</i></p> <p>The <i>CosString</i> for which the hex flag is obtained.</p> |
| Return Value | The current value of the flag. |
| Exceptions | cosErrExpectedString |
| Notifications | None |
| Header File | <i>CosCalls.h</i> |
| Related Methods | CosStringSetHexFlag |
| Availability | Plug-ins: Available if <i>PL_COS_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

CosStringSetHexFlag

```
ASBool CosStringSetHexFlag (CosObj cosObj, ASBool setHex);
```

Description Sets the hex flag of the *CosString*. The hex flag specifies whether the *CosString* should be written out as hex when writing the Cos Object to file.

Parameters

cosObj
The *CosString* for which the hex flag is set.

setHex
The value to set for the flag.

Return Value The current value of the flag (after it is set).

Exceptions [cosErrExpectedString](#)

Notifications None

Header File *CosCalls.h*

Related Methods [CosStringGetHexFlag](#)

Availability Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

CosStringValue

```
char* CosStringValue (CosObj obj, ASInt32* nBytes);
```

Description

Gets the value of string Cos object, and the string's length. The plug-in should immediately copy the returned string.

Note: The returned value is not a C-style string. Cos string objects can contain NULL chars. Standard C string-handling functions may not work as expected.

Parameters

obj

The object whose value is obtained.

nBytes

(Filled by the method) The length of *str*, in bytes.

Return Value

The value of *obj*.

Exceptions

Raises an exception if the specified object has the wrong type, or if it is an invalid object.

Notifications

None

Header File

CosCalls.h

Related Methods

[CosNewString](#)

Example

```
ASInt32 len;
char* p;
```

```
p = CosStringValue(strCos, &len);
```

Availability

Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

Encryption/Decryption

CosDecryptData

```
void CosDecryptData (void* src, ASInt32 len, void* dst,
    char* cryptData, ASInt32 cryptDataLen);
```

Description Decrypts data in a buffer using the specified encryption key. The standard Acrobat viewer encryption/decryption algorithm (RC4 from RSA Data Security, Inc.) is used.

Parameters

src
The buffer containing the data to decrypt.

len
The number of bytes in *src*.

dst
(Filled by the method) The buffer into which the decrypted data will be placed. This may point to the same location as *src*.

cryptdata
The encryption key.

cryptDataLen
Length of the encryption key, in bytes. *cryptDataLen* cannot be greater than 5 bytes.

Return Value None

Exceptions Raises [genErrNoMemory](#) if memory is exhausted. Also raises an exception if encryption encounters an internal error.

Notifications None

Header File *PICrypt.h*

Related Methods [CosEncryptData](#)

Availability Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

CosEncryptData

```
void CosEncryptData (void* src, ASInt32 len, void* dst,
    char* cryptData, ASInt32 cryptDataLen);
```

Description Encrypts data in a buffer using the specified encryption key. The standard Acrobat viewer encryption/decryption algorithm (RC4 from RSA Data Security, Inc.) is used.

Parameters

src
The buffer containing the data to encrypt.

len
The number of bytes in *src*.

dst
(Filled by the method) The buffer into which the encrypted data will be placed. This may point to the same location as *src*.

cryptdata
The encryption key.

cryptDataLen
Length of the encryption key, in bytes.
cryptDataLen cannot be greater than 5 bytes.

Return Value None

Exceptions Raises [genErrNoMemory](#) if memory is exhausted. Also raises an exception if encryption encounters an internal error.

Notifications None

Header File *PICrypt.h*

Related Methods [CosDecryptData](#)

Availability Plug-ins: Available if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PD Layer

General

PDDrawCosObjToWindow

```
void PDDrawCosObjToWindow (CosObj cosObj, void* window,
    void* displayContext, ASBool isDPS, ASFixedMatrix* matrix,
    ASFixedRect* updateRect, CancelProc cancelProc,
    void* cancelProcClientData);
```

Description

Draws the specified stream of PDF marking operators into the specified window. This method is used for platform-independent drawing of graphics and text.

Note: [PDDrawCosObjToWindow](#) changes the following:

- the current page's CTM
- the zoom factor

Note: [PDDrawCosObjToWindow](#) dirties the PDF file.

Parameters

cosObj

The stream Cos object to draw into *window*. This stream can be created using [CosNewStream](#).

The stream's dictionary must contain a *Resources* key whose value is a dictionary specifying all the resources needed to draw the Cos object (including a *ProcSet* entry). Its structure and contents are the same as for the Resources dictionary for a *Page* object. See Section 7.5 in the [Portable Document Format Reference Manual](#) for a description of a Page object's Resources dictionary.

The stream's data is a sequence of PDF marking operators. See Appendix B in the [Portable Document Format Reference Manual](#) for a description of these operators.

A pseudocode example of the stream object is:

```
<< /Length 1000 /Filter [...filters...]
  /Resources << /ProcSet [ /PDF /Text ]
  /Font <</F5 6 0 R /F9 12 0 R>> >>
>>
stream
...stream data...
endstream
```

window

Pointer to a platform-dependent window object (*WindowPtr* or *CWindowPtr* in Mac OS, *HWND* in Windows).

In Mac OS, to draw into an offscreen *GWorld*, pass *NULL* in *window* and pass the *GWorldPtr* in *displayContext*.

In Windows, to draw into an offscreen DC, pass *NULL* for *window*.

displayContext

Pointer to a platform-dependent display context structure (*GWorldPtr* in Mac OS, *hDC* in Windows).

In Mac OS, *displayContext* is ignored if *window* is non-*NULL*.

isDps

Currently unused. Always set to *false*.

matrix

Pointer to a matrix to concatenate onto the default page matrix. It is useful for scaling and for converting from page to window coordinates.

updateRect

Pointer to a rectangle, specified in user space coordinates. Any objects outside of *updateRect* will not be drawn. All objects are drawn if *updateRect* is *NULL*.

cancelProc

Procedure called periodically to check for user cancel of the drawing operation. The default cancel proc can be obtained using [*AVAppGetCancelProc*](#). May be *NULL*, in which case no cancel proc is used.

cancelProcClientData

Pointer to user-supplied data to pass to *cancelProc* each time it is called. Should be *NULL* if *cancelProc* is *NULL*.

Return Value None

Exceptions None

| | |
|------------------------|--|
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | CosNewStream PDPageDrawContentsToWindow |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020001 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDEnumDocs

```
void PDEnumDocs ( PDDocEnumProc proc, void* clientData);
```

| | |
|------------------------|--|
| Description | Enumerates the <i>PDDocs</i> that are currently open, calling a user-supplied procedure for each open document. |
| Parameters | <p><i>proc</i></p> <p>User-supplied callback to call for each open <i>PDDoc</i>. Enumeration halts if <i>proc</i> returns <i>false</i>.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>proc</i> each time it is called.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | AVAppEnumDocs PDDocOpen |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDGetHostEncoding

```
void* PDGetHostEncoding (void);
```

Description

Indicates what kind of host encoding a system uses. Allows you to determine whether a system is Roman or non-Roman. (Non-Roman is also known as CJK-capable, that is, capable of handling multibyte character sets, such as Chinese, Japanese, or Korean.)

Host encoding is a platform-dependent encoding for the host machine. For non-UNIX Roman systems, it is MacRomanEncoding in Mac OS and WinAnsiEncoding in Windows. In UNIX (except HP-UX) Roman systems, it is ISO8859-1 (ISO Latin-1); for HP-UX, it is HP-ROMAN8. See Appendix C in the [Portable Document Format Reference Manual](#) for descriptions of MacRomanEncoding and WinAnsiEncoding. For non-Roman systems, the host encoding may be a variety of encodings, which are defined by a CMap (character map). Use [PDGetHostEncoding](#) to determine if a system's host encoding is Roman or not.

Parameters

None

Return Value

0 for a Roman system; nonzero for a non-Roman system (a structure that depends on the host encoding). Users should simply test whether this value is 0 or not.

Exceptions

None

Notifications

None

Header File

PDCalls.h

Related Methods

[PDGetPDFDocEncoding](#)

Availability

Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

PDGetPDFDocEncoding

```
ASUns8** PDGetPDFDocEncoding (void);
```

Description

Gets an array describing the differences between the platform's host encoding and PDFDocEncoding.

Host encoding is a platform-dependent encoding for the host machine. For non-UNIX Roman systems, it is MacRomanEncoding in Mac OS and WinAnsiEncoding in Windows. In UNIX (except HP-UX) Roman systems, it is ISO8859-1 (ISO Latin-1); for HP-UX, it is HP-ROMAN8. See Appendix C in the [Portable Document Format Reference Manual](#) for descriptions of MacRomanEncoding, WinAnsiEncoding, and PDFDocEncoding.

Parameters

None

Return Value

An array containing 256 elements. Each element corresponds to a code point in the PDFDocEncoding, and is either *NULL* or a pointer to a string.

If the element is *NULL*, the code point refers to the same glyph in both host encoding and PDFDocEncoding. If the element is non-*NULL*, it points to a string containing the glyph name for the code point.

Exceptions

None

Notifications

None

Header File

PDCalls.h

Related Methods

[PDGetHostEncoding](#)

Availability

Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDHostMBLen

```
ASInt32 PDHostMBLen (const char* cp);
```

Description

Gets the number of additional bytes required for the multibyte character pointed to by *cp*. If *cp* points to a single-byte character, 0 is returned. This method allows determining the length of multibyte character strings to allocate space for them.

PDHostMBLen has similar functionality to the ANSI-C code:

```
mblen(cp, MB_LEN_MAX) - 1
```

or the Windows function:

```
IsDBCSLeadByte(cp)
```

Parameters

cp

The character to examine.

Return Value

The number of bytes in the multibyte character.

Exceptions

None

Notifications

None

Header File

PDCalls.h

Related Methods

None

Example

```
char* strchrMB(char* cp, char* ch) {
    ASInt32 ch_len = PDHostMBLen(ch);

    while (*cp != '\0') {
        ASInt32 cp_len = PDHostMBLen(cp);

        if (ch_len == cp_len) {
            ASInt32 i;

            for (i = 0; i < ch_len; i++)
                if (cp[i] != ch[i])
                    break;

            if (i == ch_len)
                return cp;
        }

        cp += (cp_len + 1);
    }

    return NULL;
}
```

Availability

Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDFPrefGetColorCal

```
ASBool PDFPrefGetColorCal (PDColorCalP colorCal);
```

Description

Gets the values to use for displaying calibrated color and grayscale. These values are the chromaticities and gammas of the phosphors in the monitor.

These values do not necessarily correspond to the monitor being used; it is the responsibility of the plug-in that sets these values to provide the correct values.

These values are used for rendering if calibrated color is enabled by the preferences file item *avpDoCalibratedColor* (see [AVAppSetPreference](#)).

Parameters

colorCal

(Filled by the method) Pointer to a structure that contains the color calibration information.

For RGB devices, the red, green, and blue chromaticities and gammas are used; for grayscale, *whiteChrom* and *greenGamma* are used.

You must allocate storage for the *colorCal* data structure and pass a pointer to the memory you allocated.

Return Value

Always returns *true*.

Exceptions

None

Notifications

None

Header File

PDCalls.h

Related Methods

[PDFPrefSetColorCal](#)
[AVAppSetPreference](#)

Availability

Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

PDFPrefSetColorCal

```
ASBool PDFPrefSetColorCal (PDCColorCalP colorCal);
```

Description

Sets the values to use for displaying calibrated color and grayscale. These values are the chromaticities and gammas of the phosphors in the monitor.

These values do not necessarily correspond to the monitor being used; it is the responsibility of the plug-in that sets these values to provide the correct values.

These values are used for rendering if calibrated color is enabled by the preferences file item *avpDoCalibratedColor* (see [AVAppSetPreference](#)).

Parameters

colorCal

Pointer to a structure that contains the color calibration information.

For RGB devices, the red, green, and blue chromaticities and gammas are used; for grayscale, *whiteChrom* and *greenGamma* are used.

Return Value

true if the values were successfully set and installed for the currently active display device, *false* otherwise.

Exceptions

None

Notifications

None

Header File

PDCalls.h

Related Methods

[PDFPrefGetColorCal](#)
[AVAppSetPreference](#)

Availability

Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDRegisterCryptHandler

```
void PDRegisterCryptHandler ( PDCryptHandler handler,
    const char* pdfName, const char* userName);
```

| | |
|------------------------|---|
| Description | Registers a new security handler with the Acrobat viewer. |
| Parameters | <p><i>handler</i></p> <p>Pointer to a structure that contains the security handler's callback functions. This structure <i>must not</i> be freed after calling <i>PDRegisterCryptHandler</i>.</p> <p><i>pdfName</i></p> <p>The name of the security handler as it will appear in the PDF file. This name is also used by PDDocSetNewCryptHandler. Storage for this name can be freed after <i>PDRegisterCryptHandler</i> has been called.</p> <p><i>userName</i></p> <p>The name of the security handler as it will be shown in menus. This name can be localized into different languages. Storage for this name can be freed after <i>PDRegisterCryptHandler</i> has been called.</p> |
| Return Value | None |
| Exceptions | <p>Raises genErrBadParm if the security handler's <i>size</i> field is incorrect.</p> <p>Raises pdModErrDuplicateCryptName if either <i>pdfName</i> or <i>userName</i> are already in use by a registered security handler.</p> <p>Raises genErrNoMemory if memory is exhausted.</p> <p>May raise other exceptions.</p> |
| Notifications | None |
| Header File | <i>PICrypt.h</i> |
| Related Methods | PDDocSetNewCryptHandler PDRegisterCryptHandlerEx |

Example

```

PDCryptHandler handler;

handler = (PDCryptHandler) ASmallocc(
    sizeof(PDCryptHandlerRec));
handler->size = sizeof(PDCryptHandlerRec);
handler->NewAuthData = 0;
handler->GetAuthData = 0;
handler->Authorize =
    ASCallbackCreateProto(
        PDCryptAuthorizeProc, &Authorize);
handler->NewSecurityData =
    ASCallbackCreateProto(
        PDCryptNewSecurityDataProc,
        &NewSecurityData);
handler->ValidateSecurityData =
    ASCallbackCreateProto(
        PDCryptValidateSecurityDataProc,
        &ValidateSecurityData);
handler->UpdateSecurityData =
    ASCallbackCreateProto(
        PDCryptUpdateSecurityDataProc,
        &UpdateSecurityData);
handler->NewCryptData =
    ASCallbackCreateProto(
        PDCryptNewCryptDataProc, &NewCryptData);
handler->FillEncryptDict =
    ASCallbackCreateProto(
        PDCryptFillEncryptDictProc,
        &FillEncryptDict);
handler->GetSecurityInfo =
    ASCallbackCreateProto(
        PDCryptGetSecurityInfoProc,
        &GetSecurityInfo);
DURING
    PDRegisterCryptHandler(handler,
        "Subscription", "Subscript");
HANDLER
...
END_HANDLER

```

Availability

Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDRegisterCryptHandlerEx

```
void PDRegisterCryptHandlerEx (PDCryptHandler handler,
    const char* pdfName, const char* userName,
    void* clientData);
```

| | |
|----------------------|---|
| Description | Registers a new security handler with the Acrobat viewer. Same as PDRegisterCryptHandler except that it accepts a client data parameter. |
| Parameters | <p><i>handler</i></p> <p>Pointer to a structure that contains the security handler's callback functions. This structure must <i>not</i> be freed after calling PDRegisterCryptHandlerEx.</p> <p><i>pdfName</i></p> <p>The name of the security handler as it will appear in the PDF file. This name is also used by PDDocSetNewCryptHandler. Storage for this name can be freed after PDRegisterCryptHandlerEx has been called.</p> <p><i>userName</i></p> <p>The name of the security handler as it will be shown in menus. This name can be localized to different languages. Storage for this name can be freed after PDRegisterCryptHandlerEx has been called.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to store with the handler.</p> |
| Return Value | None |
| Exceptions | <p>Raises genErrBadParm if the security handler's <i>size</i> field is incorrect.</p> <p>Raises pdModErrDuplicateCryptName if either <i>pdfName</i> or <i>userName</i> are already in use by a registered security handler.</p> <p>Raises genErrNoMemory if memory is exhausted.</p> <p>May raise other exceptions.</p> |
| Notifications | None |
| Header File | <i>PICrypt.h</i> |

Related Methods [PDDocSetNewCryptHandler](#)
[PDRegisterCryptHandler](#)

Example

```

PDCryptHandler handler;

handler = (PDCryptHandler) ASmallocc(
    sizeof(PDCryptHandlerRec));
handler->size = sizeof(PDCryptHandlerRec);
handler->NewAuthData = 0;
handler->GetAuthData = 0;
handler->Authorize =
    ASCallbackCreateProto(
        PDCryptAuthorizeProc, &Authorize);
handler->NewSecurityData =
    ASCallbackCreateProto(
        PDCryptNewSecurityDataProc,
        &NewSecurityData);
handler->ValidateSecurityData =
    ASCallbackCreateProto(
        PDCryptValidateSecurityDataProc,
        &ValidateSecurityData);
handler->UpdateSecurityData =
    ASCallbackCreateProto(
        PDCryptUpdateSecurityDataProc,
        &UpdateSecurityData);
handler->NewCryptData =
    ASCallbackCreateProto(
        PDCryptNewCryptDataProc, &NewCryptData);
handler->FillEncryptDict =
    ASCallbackCreateProto(
        PDCryptFillEncryptDictProc,
        &FillEncryptDict);
handler->GetSecurityInfo =
    ASCallbackCreateProto(
        PDCryptGetSecurityInfoProc,
        &GetSecurityInfo);
DURING
    PDRegisterCryptHandler(handler,
        "Subscription", "Subscript");
HANDLER
...
END_HANDLER
    
```

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDRegisterFileSpecHandler

```
void PDRegisterFileSpecHandler (ASFileSys contextFileSys,
    PDFileSpecHandler fileSpecHandler, void* fileSpecHandlerObj);
```

Description Registers a new file specification handler with the Acrobat viewer. In version 3.0 and later of the Acrobat viewer, use the [PDRegisterFileSpecHandlerByName](#) method instead.

Parameters *contextFileSys*

The file system that specifies the context in which the file specification handler is used. This is the file system on which the PDF document resides. It is sometimes necessary to use different file specification handlers depending on the file system in which the document is open. For example, when a document is opened in a Web browser, the Acrobat viewer may use the browser's http stack when it needs to use http. When a document is opened outside of the browser, however, the Acrobat viewer must use a different http stack.

fileSpecHandler

Pointer to a structure that contains the handler's callbacks. This structure *must* not be freed after calling *PDRegisterFileSpecHandler*.

fileSpecHandlerObj

Pointer to user-supplied data to pass to the file specification handler's callbacks each time they are called.

Return Value None

Exceptions [genErrNoMemory](#)

Notifications None

Header File *PDCalls.h*

Related Methods [PDRegisterFileSpecHandlerByName](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDRegisterFileSpecHandlerByName

```
void PDRegisterFileSpecHandlerByName (ASAtom specSysName,  
    ASFileSys contextFileSys, PDFileSpecHandler fileSpecHandler,  
    void* fileSpecHandlerObj);
```

Description

Registers a new file specification handler with the Acrobat viewer. The viewer calls the appropriate file specification handler when it encounters a file specification in a PDF file. The appropriate file specification handler is the one whose:

- *specSysName* matches the value of the /FS key in the file specification,
- *contextFileSys* matches the file system on which the PDF file resides.

The file specification handler's file system, (passed as the *fileSys* field of *fileSpecHandler*), is used to obtain data from, or write data to, the file referred to by the file specification.

Parameters

specSysName

The name (as an *ASAtom*) of a file system with which this file specification works.

contextFileSys

The file system that specifies the context in which the file specification handler is used. This is the file system on which the PDF document resides. It is sometimes necessary to use different file specification handlers depending on the file system in which the document is open. For example, when a document is opened in a Web browser, the Acrobat viewer may use the browser's http stack when it needs to use http. When a document is opened outside of the browser, however, the Acrobat viewer must use a different http stack.

fileSpecHandler

Pointer to a structure that contains the handler's callbacks. This structure *must* not be freed after calling *PDRegisterFileSpecHandlerByName*.

fileSpecHandlerObj

Pointer to user-supplied data to pass to the file specification handler's callbacks each time they are called.

Return Value None

Exceptions [genErrNoMemory](#)

Notifications None

Header File *PDCalls.h*

Related Methods [PDRegisterFileSpecHandler](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDSetHostEncoding

```
void PDSetHostEncoding (void* encoding, char* parseTable);
```

| | |
|------------------------|--|
| Description | (<i>UNIX only</i>) Sets the host encoding. The only currently supported host encoding is Japanese: EUC-JA. Do <i>not</i> call this method to extract Roman-only text. |
| Parameters | <i>encoding</i> Host encoding. <i>parseTable</i> Pathname to which PostScript file is printed. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Example | PDFToolkitTerm(); |
| Related Methods | PDFLInit |
| Availability | Library: Available if <i>kPDFLVersion</i> (obtained from <i>PDFLGetVersion</i>) is 0x00010000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

PDXlateToHost

```
void PDXlateToHost (char* in, char* out, ASInt32 numBytes);
```

Description

Translates a string from PDFDocEncoding to host encoding. This method is useful when setting or retrieving displayed text that *must* be in PDFDocEncoding (or Unicode), such as text that appears in a text annotation or bookmark.

A character that cannot be converted to the destination encoding is replaced with a space.

Host encoding is a platform-dependent encoding for the host machine. For non-UNIX Roman systems, it is MacRomanEncoding in Mac OS and WinAnsiEncoding in Windows. In UNIX (except HP-UX) Roman systems, it is ISO8859-1 (ISO Latin-1); for HP-UX, it is HP-ROMAN8. See Appendix C in the [Portable Document Format Reference Manual](#) for descriptions of MacRomanEncoding, WinAnsiEncoding, and PDFDocEncoding.

For non-Roman systems, the host encoding may be a variety of encodings, which are defined by a CMap (character map). See Section 7.9 in the [Portable Document Format Reference Manual](#) for a list of predefined CMaps. Use [PDGetHostEncoding](#) to determine if a system's host encoding is Roman or not. For non-Roman systems, use [PDXlateToHostEx](#).

In general, [PDXlateToHostEx](#) can be called instead of *PDXlateToHost* since [PDXlateToHostEx](#) works for any host encoding.

Parameters

in

The string to translate (may point to the same memory as *out*, allowing strings to translate in place).

out

(*Filled by the method*) The translated string (may point to the same memory as *in*).

numBytes

Number of bytes in the string to translate.

Return Value

None

| | |
|------------------------|--|
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDGetHostEncoding PDXlateToHostEx PDXlateToPDFDocEnc PDXlateToPDFDocEncEx |

Example

```
ASUns16 attr;
attr = PDWordGetAttr(myWord);
wlen = PDWordGetLength(myWord);
if(attr & WXE_HAS_NONALPHANUM)/* handle
    Mac quotes */
    PDXlateToPDFDocEnc(&msg[0], &msg[0],
        strlen(msg));
```

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDXlateToHostEx

```
ASInt32 PDXlateToHostEx (const char* inPdfStr,  
    ASInt32 inPdfStrSize, char* outHostStr,  
    ASInt32 outHostStrSize);
```

Description

Translates a string from Unicode or PDFDocEncoding to host encoding. This method is useful when setting or retrieving displayed text that might be in Unicode, such as text that appears in a text annotation or bookmark.

A character that cannot be converted to the destination encoding is replaced with a space.

Host encoding is a platform-dependent encoding for the host machine. For non-UNIX Roman systems, it is MacRomanEncoding in Mac OS and WinAnsiEncoding in Windows. In UNIX (except HP-UX) Roman systems, it is ISO8859-1 (ISO Latin-1); for HP-UX, it is HP-ROMAN8. See Appendix C in the [Portable Document Format Reference Manual](#) for descriptions of MacRomanEncoding, WinAnsiEncoding, and PDFDocEncoding. For non-Roman systems, the host encoding may be a variety of encodings, which are defined by a CMap (character map). See Section 7.9 in the [Portable Document Format Reference Manual](#) for information on CMaps.

For non-Roman systems, use *PDXlatetoHostEx*. Use [PDGetHostEncoding](#) to determine whether the host encoding for a system is Roman or not.

In general, *PDXlatetoHostEx* operates the same as [PDXlateToHost](#) but requires an extra argument, since the sizes of the input and translated strings may differ. This method can be called instead of [PDXlateToHost](#)—and *must* be called for multibyte character set systems.

Parameters

inPdfStr

Pointer to the string to translate (may point to the same memory as *outHostStr*, allowing strings to translate in place).

inPdfStrSize

The length of *inPdfStr*, in bytes.

outHostStr

(Filled by the method) Pointer to the translated string (may point to the same memory as *inPdfStr*).

outHostStrSize

The length of the *outHostStr* buffer, in bytes.

Return Value Number of bytes in the translated string *outHostStr*.

Exceptions None

Notifications None

Header File *PDCalls.h*

Related Methods [PDFontXlateToHost](#)
[PDFontXlateToUCS](#)
[PDGetHostEncoding](#)
[PDXlateToHost](#)
[PDXlateToPDFDocEnc](#)
[PDXlateToPDFDocEncEx](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDXlateToPDFDocEnc

```
void PDXlateToPDFDocEnc (char* in, char* out,
    ASInt32 numBytes);
```

Description

Translates a string from host encoding to PDFDocEncoding. This method is useful when setting or retrieving displayed text that *must* be in PDFDocEncoding (or Unicode), such as text that appears in a text annotation or bookmark.

A character that cannot be converted to the destination encoding is replaced with a space. For example, *PDXlateToPDFDocEnc* converts *\r* to a space character (*\r* is present in PDFDocEncoding and is left unchanged).

Host encoding is a platform-dependent encoding for the host machine. For non-UNIX Roman systems, it is MacRomanEncoding in Mac OS and WinAnsiEncoding in Windows. In UNIX (except HP-UX) Roman systems, it is ISO8859-1 (ISO Latin-1); for HP-UX, it is HP-ROMAN8. See Appendix C in the [Portable Document Format Reference Manual](#) for descriptions of MacRomanEncoding, WinAnsiEncoding, and PDFDocEncoding.

For non-Roman systems, the host encoding may be a variety of encodings, which are defined by a CMap (character map). See Section 7.9 in the [Portable Document Format Reference Manual](#) for a list of predefined CMaps. Use [PDGetHostEncoding](#) to determine if a system's host encoding is Roman or not. For non-Roman systems, use [PDXlateToPDFDocEncEx](#).

In general, [PDXlateToPDFDocEncEx](#) can be called instead of *PDXlateToPDFDocEnc* since [PDXlateToPDFDocEncEx](#) works for PDFDocEncoding or Unicode.

Parameters

in

The string to translate (may point to the same memory as *out*, allowing strings to translate in place).

out

(Filled by the method) The translated string (may point to the same memory as *in*).

numBytes

Number of bytes in the string to translate.

Return Value None

Exceptions None

Notifications None

Header File *PDCalls.h*

Related Methods [PDGetHostEncoding](#)
[PDXlateToHost](#)
[PDXlateToHostEx](#)
[PDXlateToPDFDocEncEx](#)

Example

```
ASUns16 attr;
attr = PDWordGetAttr(myWord);
wlen = PDWordGetLength(myWord);
if(attr & WXE_HAS_NONALPHANUM)/* handle
    Mac quotes */
    PDXlateToPDFDocEnc(&msg[0], &msg[0],
        strlen(msg));
```

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDXlateToPDFDocEncEx

```
ASInt32 PDXlateToPDFDocEncEx (ASBool bUseUnicode,  
    const char* inHostStr, ASInt32 inHostStrSize,  
    char* outPDFStr, ASInt32 outPDFStrSize);
```

Description

Translates a string from host encoding to PDFDocEncoding or Unicode. This method is useful when using text that must be in PDFDocEncoding or Unicode, such as text in a text annotation, bookmark, or article title.

A character that cannot be converted to the destination encoding is replaced with a space. For example, *PDXlateToPDFDocEncEx* converts *ln* to a space character (*l* is present in PDFDocEncoding and is left unchanged).

Host encoding is a platform-dependent encoding for the host machine. For non-UNIX Roman systems, it is MacRomanEncoding in Mac OS and WinAnsiEncoding in Windows. In UNIX (except HP-UX) Roman systems, it is ISO8859-1 (ISO Latin-1); for HP-UX, it is HP-ROMAN8. See Appendix C in the [Portable Document Format Reference Manual](#) for descriptions of MacRomanEncoding, WinAnsiEncoding, and PDFDocEncoding.

For non-Roman systems, the host encoding may be a variety of encodings, which are defined by a CMap (character map). See Section 7.9 in the [Portable Document Format Reference Manual](#) for a list of predefined CMaps.

For non-Roman systems, use *PDXlateToPDFDocEncEx*. You can use [PDGetHostEncoding](#) to determine whether a system's host encoding is Roman or not.

In general, *PDXlateToPDFDocEncEx* can be called instead of [PDXlateToPDFDocEnc](#) since *PDXlateToPDFDocEncEx* works for PDFDocEncoding or Unicode.

Parameters

bUseUnicode

If *true*, translate the string to Unicode; otherwise use PDFDocEncoding.

inHostStr

Pointer to the string to translate (may point to the same memory as *outPDFStr*, allowing strings to translate in place).

inHostStrSize

Number of bytes in the string to translate.

outPDFStr

(Filled by the method) Pointer to the translated string (may point to the same memory as *inHostStr*).

outPDFStrSize

The length of the *outPDFStr* buffer, in bytes.

Return Value Number of bytes in the translated string *outPDFStr*.

Exceptions None

Notifications None

Header File *PDCalls.h*

Related Methods [PDFontXlateToHost](#)
[PDFontXlateToUCS](#)
[PDGetHostEncoding](#)
[PDXlateToHost](#)
[PDXlateToHostEx](#)
[PDXlateToPDFDocEnc](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDAction

PDActionDestroy

```
void PDActionDestroy (PDAction action);
```

Description Destroys an action object.

Parameters *action*
The action to destroy.

Return Value None

Exceptions None

Notifications None

Header File *PDCalls.h*

Related Methods [PDActionNew](#)
[PDActionNewFromDest](#)
[PDActionNewFromFileSpec](#)
[PDActionFromCosObj](#)

Example

```
if(PDActionIsValid(oldAction)
    PDActionDestroy(oldAction);
```

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDActionEqual

```
ASBool PDActionEqual (PDAction action, PDAction action2);
```

Description Compares two actions for equality. Two actions are equal if and only if their Cos objects are equal (see [CosObjEqual](#)).

Parameters *action, action2*
The two actions that are compared.

Return Value *true* if the actions are equal, *false* otherwise.

Exceptions None

Notifications None

Header File *PDCalls.h*

Related Methods [CosObjEqual](#)

Example

```
if(PDActionIsValid(oldAction) &&
    PDActionIsValid(newAction) &&
    PDActionEqual(oldAction, newAction))
    PDActionDestroy(oldAction);
```

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDActionFromCosObj

[PDAction](#) PDActionFromCosObj ([CosObj](#) obj);

Description Converts a dictionary Cos object to an action and verifies that the action is valid. This method does not copy the object, but is instead the logical equivalent of a type cast.

Parameters *action*
Dictionary Cos object for the action.

Return Value The *PDAction* corresponding to *obj*.

Exceptions Raises [pdErrBadAction](#) if the action is invalid as determined by [PDActionIsValid](#).

Notifications None

Header File *PDCalls.h*

Related Methods [PDActionGetCosObj](#)
[PDActionNew](#)
[PDActionNewFromDest](#)
[PDActionNewFromFileSpec](#)

Example
CosObj theObj;
action = PDActionFromCosObj(cosObj);

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDActionGetCosObj

[CosObj](#) PDActionGetCosObj ([PDAction](#) action);

| | |
|------------------------|---|
| Description | Gets the Cos object corresponding to an action. This method does not copy the object, but is instead the logical equivalent of a type cast. |
| Parameters | <p><i>action</i></p> <p>The action whose Cos object is obtained.</p> |
| Return Value | Dictionary Cos object for the action. The contents of the dictionary can be enumerated using CosObjEnum . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDActionFromCosObj |
| Example | <pre> CosObj theObj; if (PDActionGetSubtype(act) == ASAtomFromString("Thread")) theObj = PDActionGetCosObj(act); </pre> |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDActionGetDest

[PDViewDestination](#) PDActionGetDest ([PDAction](#) action);

| | |
|------------------------|--|
| Description | <p>Gets an action's destination view. This only works for actions that contain a view destination, that is, actions whose subtype is <i>GoTo</i>.</p> <p>For named destinations, this method may return a Cos string object or a Cos name object. See Section 7.3.2 in the Portable Document Format Reference Manual for more information on named destinations.</p> <p><i>Note:</i> Since this method may not return a PDViewDestination, use the PDViewDestResolve method on the returned value to obtain a PDViewDestination.</p> |
| Parameters | <p><i>action</i></p> <p>The action whose destination is obtained.</p> |
| Return Value | <p>The action's destination, which may be either a PDViewDestination—or for named destinations—a Cos string object or a Cos name object. Use the PDViewDestResolve method on this returned value to obtain a PDViewDestination.</p> |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDActionGetFileSpec PDViewDestResolve |
| Example | <pre>PDViewDestination pdViewDest; PDViewDestination pdViewDestFinal; PDDoc pdDoc; pdViewDest = PDActionGetDest(action); pdViewDestFinal = PDViewDestResolve(pdViewDest, pdDoc);</pre> |
| Availability | <p>Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher.</p> |
| Available in: | |

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDActionGetFileSpec

```
PDFFileSpec PDActionGetFileSpec (PDAction action);
```

| | |
|------------------------|---|
| Description | Gets a file specification from an action. Not all types of actions have file specifications, and this method only work for actions that contain a file specification. See Section 6.8 in the Portable Document Format Reference Manual for more information on the contents of various types of actions. |
| Parameters | <i>action</i> The action whose file specification is obtained. |
| Return Value | The action's file specification. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDActionGetDest |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDActionGetSubtype

```
ASAtom PDActionGetSubtype (PDAction action);
```

| | |
|------------------------|--|
| Description | Gets an action's subtype. |
| Parameters | <i>action</i> The action whose subtype is obtained. |
| Return Value | The <i>ASAtom</i> corresponding to the action's subtype. The <i>ASAtom</i> can be converted to a string using ASAtomGetCount . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | None |
| Example | <pre>CosObj theObj; if(PDActionGetSubtype(act) == ASAtomFromString("Thread")) theObj = PDActionGetCosObj(act);</pre> |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDActionIsValid

```
ASBool PDActionIsValid (PDAction action);
```

Description Tests an action's validity. This is intended only to ensure that the action has not been deleted, not to ensure that all necessary information is present and valid.

Parameters *action*
The action whose validity is determined.

Return Value *true* if the action is valid, *false* otherwise.

Exceptions None

Notifications None

Exceptions None

Header File *PDCalls.h*

Related Methods [PDActionEqual](#)

Example

```
if (PDActionIsValid(oldAction)
    PDActionDestroy(oldAction);
```

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDActionNew

```
PDAction PDActionNew (PDDoc doc, ASAtom type);
```

| | |
|------------------------|---|
| Description | Creates a new action object. |
| Parameters | <p><i>doc</i></p> <p>The document in which the action is created.</p> <p><i>type</i></p> <p>The <i>ASAtom</i> corresponding to the action's subtype. The <i>ASAtom</i> can be obtained from a string using <i>ASAtomFromString</i>.</p> |
| Return Value | The newly-created <i>PDAction</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDActionNewFromDest PDActionNewFromFileSpec PDActionFromCosObj ASAtomFromString PDActionDestroy |
| Example | <pre>#define Thread_K ASAtomFromString("Thread") PDAction myAct = PDActionNew(doc, Thread_K);</pre> |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDActionNewFromDest

```
PDAction PDActionNewFromDest (PDDoc doc, PDViewDestination dest,
PDDoc destDoc);
```

| | |
|------------------------|--|
| Description | Creates a new action that takes the user to the specified destination view. This method can only be used for destinations in the same document as the source document. Cross-document links must be built up from the Cos level, populating the Action dictionary for the <i>GotoR</i> action as described in Section 6.8.3 in the Portable Document Format Reference Manual . |
| Parameters | <p><i>doc</i></p> <p>The document in which the action is created and used.</p> <p><i>dest</i></p> <p>The destination view.</p> <p><i>destDoc</i></p> <p>Destination document. <i>destDoc</i> must be the same as <i>doc</i>.</p> |
| Return Value | The newly-created action. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDActionNew PDActionNewFromFileSpec PDActionFromCosObj PDActionDestroy |
| Example | <pre>dstPage = PDDocAcquirePage(pdDoc, (offset + Pages[i].pg)-1); dest = PDViewDestCreate(pdDoc, dstPage, fit, &initRect, zoom, Pages[i].pg-1); if(PDViewDestIsValid(dest)){ pdAction = PDActionNewFromDest(pdDoc, dest, pdDoc); PDBookmarkSetAction(newBm, pdAction); }</pre> |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDActionNewFromFileSpec

```
PDAction PDActionNewFromFileSpec (PDDoc containingDoc,  
    ASAtom type, PDFFileSpec fileSpec);
```

| | |
|------------------------|--|
| Description | Creates an action of the specified type from a file specification. |
| Parameters | <i>containingDoc</i> The document in which the action is created and used. <i>type</i> The type of action to create. <i>fileSpec</i> The file specification that is made part of an action. |
| Return Value | The newly-created <i>PDAction</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDActionNew PDActionNewFromDest PDActionFromCosObj PDActionDestroy |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDAnnot

PDAnnotEqual

```
ASBool PDAnnotEqual (PDAnnot anAnnot, PDAnnot annot2);
```

| | |
|------------------------|--|
| Description | Tests whether or not two annotations are identical. |
| Parameters | <i>anAnnot</i> , <i>annot2</i> The two annotations to compare. |
| Return Value | <i>true</i> if the annotations are equal, <i>false</i> otherwise. Two annotations are equal if and only if their Cos objects are equal (see CosObjEqual). |
| Exceptions | pdErrBadAnnotation |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | CosObjEqual |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDAnnotFromCosObj

[PDAnnot](#) PDAnnotFromCosObj ([CosObj](#) obj);

| | |
|------------------------|--|
| Description | Converts a dictionary Cos object to an annotation. This method does not copy the object, but is instead the logical equivalent of a type cast. |
| Parameters | <i>obj</i> Dictionary Cos object for the annotation. |
| Return Value | The <i>PDAnnot</i> corresponding to the Cos object. |
| Exceptions | Raises pdErrBadAnnotation if the annotation is invalid, as determined by PDAnnotIsValid . |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDAnnotGetCosObj |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDAnnotGetColor

```
ASBool PDAnnotGetColor (PDAnnot anAnnot, PDColorValue color);
```

| | |
|------------------------|--|
| Description | Gets a note or link annotation's color. If the annotation does not specify an explicit color, a default color is returned. Text annotations return "default yellow;" all others (links, and so forth) return black. Only RGB color specifications are currently supported. |
| Parameters | <p><i>anAnnot</i></p> <p>The note or link annotation whose color is obtained.</p> <p><i>color</i></p> <p><i>(Filled by the method)</i> The annotation's color, which is used as follows:</p> <ul style="list-style-type: none">• Closed text note—the icon background color• Open, unselected text note—bounding rectangle color• Open, selected text note—color of annotation's title bar• Link annotation—link border color |
| Return Value | <i>true</i> if the annotation specifies an explicit color, <i>false</i> if a default color was used. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDAnnotSetColor PDAnnotGetDate PDAnnotGetFlags PDAnnotGetRect PDAnnotGetTitle |

Example

```
for(i=0;i<PDPageGetNumAnnots(pdPage);i++){
    annot = PDPageGetAnnot(pdPage, i);
    if(PDAnnotIsValid(annot) &&
        PDAnnotGetSubtype(annot) ==
            ASAtomFromString("Text"))
    {
        if(!PDAnnotGetColor(annot, oldColor))
            PDAnnotSetColor(annot, newColor);
    }
}
```

Availability

Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDAnnotGetCosObj

[CosObj](#) PDAnnotGetCosObj ([PDAnnot](#) annot);

| | |
|------------------------|--|
| Description | Gets the Cos object corresponding to an annotation. This method does not copy the object, but is instead the logical equivalent of a type cast. |
| Parameters | <p><i>annot</i></p> <p>The annotation whose Cos object is obtained.</p> |
| Return Value | Dictionary Cos object for the annotation. The contents of the dictionary can be enumerated using CosObjEnum . Returns a <i>NULL</i> Cos object if the annotation is not valid, as determined by PDAnnotIsValid . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDAnnotFromCosObj |

Example

```
for (i=0;i<PDPageGetNumAnnots(pdPage);i++)
{
    annot = PDPageGetAnnot(pdPage, i);
    if(PDAnnotIsValid(annot) &&
        PDAnnotGetSubtype(annot) ==
        ASAtomFromString("Link"))
    {
        CosObj myCos = PDAnnotGetCosObj(
            annot);
        /* now process myCos */
        ...
    }
}
```

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDAnnotGetDate

```
ASBool PDAnnotGetDate ( PDAnnot anAnnot, ASTimeRecP date );
```

| | |
|------------------------|--|
| Description | Gets an annotation's date. |
| Parameters | <p><i>anAnnot</i></p> <p>The annotation whose date is obtained.</p> <p><i>date</i></p> <p>(Filled by the method) The annotation's time and date.</p> |
| Return Value | <i>true</i> if the annotation contains a date key and the value of that key can be successfully parsed as a date string, <i>false</i> otherwise. |
| Exceptions | <p>Raises pdErrBadAnnotation if the annotation is not valid or if the value of the annotation's M (ModDate) key is not a string.</p> <p>Raises if genErrBadParm <i>date</i> is <i>NULL</i>.</p> |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDAnnotSetDate PDAnnotGetColor PDAnnotGetFlags PDAnnotGetRect PDAnnotGetTitle |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDAnnotGetFlags

```
ASUns32 PDAnnotGetFlags (PDAnnot anAnnot);
```

| | |
|------------------------|--|
| Description | Gets an annotation's flags. |
| Parameters | <p><i>anAnnot</i></p> <p>The annotation whose flags are obtained.</p> <p><i>flags</i></p> <p>(Filled by the method) An OR of the PDAnnot Flags values.</p> |
| Return Value | The flags, or 0 if the annotation does not have a <i>flags</i> key. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDAnnotSetFlags PDAnnotGetColor PDAnnotGetDate PDAnnotGetRect PDAnnotGetTitle |
| Example | <pre>if (!PDAnnotGetFlags(annot) & pdAnnotInvisible) PDAnnotSetFlags(annot, pdAnnotInvisible);</pre> |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDAnnotGetRect

```
void PDAnnotGetRect (PDAnnot anAnnot, ASFixedRect* boxP);
```

Description Gets the size and location of an annotation on its page.

Parameters *anAnnot*

The annotation whose location and size are set.

boxP

(Filled by the method) Pointer to a rectangle that specifies the annotation's bounding rectangle, specified in user space coordinates.

Return Value None

Exceptions [pdErrBadAnnotation](#)

Notifications None

Header File *PDCalls.h*

Related Methods [PDAnnotSetRect](#)
[PDAnnotGetColor](#)
[PDAnnotGetDate](#)
[PDAnnotGetFlags](#)
[PDAnnotGetTitle](#)

Example

```
ASFixedRect frect;
for(i=0;i<PDPageGetNumAnnots(pdPage);i++){
    annot = PDPageGetAnnot(pdPage, i);
    if(PDAnnotIsValid(annot) &&
        PDAnnotGetSubtype(annot) ==
        ASAtomFromString("Text"))
    {
        PDAnnotGetRect(annot, &frect);
        /* Modify the annot's rect */
        ...
        /* Set the annot's rect */
        PDAnnotSetRect(annot, &frect);
    }
}
```

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDAnnotGetSubtype

[ASAtom](#) PDAnnotGetSubtype ([PDAnnot](#) anAnnot);

| | |
|------------------------|---|
| Description | Gets an annotation's subtype. |
| Parameters | <i>anAnnot</i> The annotation whose subtype is obtained. |
| Return Value | The <i>ASAtom</i> for the annotation's subtype. This can be converted to a string using ASAtomGetCount . The storage pointed to by the return value is owned by the Acrobat viewer and should be assumed to be valid only until the next call into <i>any</i> plug-in API method; it should be immediately copied by the plug-in if the plug-in wants to reference it later. Returns <i>ASAtomNull</i> if the annotation does not have a <i>Subtype</i> key or its value is not a name object. |
| Exceptions | pdErrBadAnnotation |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | None |
| Example | <pre>for (i=0;i<PDPageGetNumAnnots(pdPage);i++) { annot = PDPageGetAnnot(pdPage, i); if(PDAnnotIsValid(annot) && PDAnnotGetSubtype(annot) == ASAtomFromString("Link")) { ... } }</pre> |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDAnnotGetTitle

```
ASInt32 PDAnnotGetTitle (PDAnnot anAnnot, char* buffer,  
    ASInt32 bufSize);
```

Description

Gets an annotation's label text.

Note: For Roman viewers, this text is always stored in the PDFDocEncoding. For non-Roman character set viewers, this text is stored as PDFDocEncoding or Unicode, depending on the file's creator. Files created in a non-Roman environment contain Unicode versions of these strings; in a Roman environment, files contain PDFDocEncoding versions of these strings.

Parameters

anAnnot

The annotation whose label is obtained.

buffer

(Filled by the method) String into which the annotation's label is copied. If *str* is non-*NULL*, up to *bufSize* bytes are copied into *buffer*.

bufSize

Buffer size, in bytes. Up to *bufSize* bytes of the annotation label string into *str* and appends an ASCII *NULL* character. The caller is expected to have allocated *bufSize* + 1 bytes to allow for the *NULL*. If *buffer* is *NULL*, copies nothing.

Return Value

If *str* is non-*NULL*, the number of bytes copied, not counting the trailing *NULL*. If *str* is *NULL*, the number of bytes that would be copied if *str* were not *NULL*.

Exceptions

[pdErrBadAnnotation](#)

Notifications

None

Header File

PDCalls.h

Related Methods

[PDAnnotSetTitle](#)
[PDAnnotGetColor](#)
[PDAnnotGetDate](#)
[PDAnnotGetRect](#)
[PDAnnotGetFlags](#)

Example

```
for(i=0;i<PDPageGetNumAnnots(pdPage);i++){
    annot = PDPageGetAnnot(pdPage, i);
    if(PDAnnotIsValid(annot) &&
        PDAnnotGetSubtype(annot) ==
        ASAtomFromString("Text"))
    {
        if(!PDAnnotGetTitle(annot, buf,
            sizeof(buf)))
        {
            strcat(buf, "-RD");
            PDAnnotSetTitle(annot, buf,
                sizeof(buf));
        }
    }
}
```

Availability

Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDAnnotIsValid

```
ASBool PDAnnotIsValid (PDAnnot anAnnot);
```

| | |
|------------------------|--|
| Description | Tests whether or not an annotation is valid. This is intended only to ensure that the annotation has not been deleted, not to ensure that all necessary information is present and valid. |
| Parameters | <p><i>anAnnot</i></p> <p>The annotation whose validity is tested.</p> |
| Return Value | <i>true</i> if <i>annot</i> is a valid annotation object, <i>false</i> otherwise. An annotation is valid if it is a Cos dictionary object and has a <i>Rect</i> key. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDAnnotEqual |
| Example | <pre>for (i=0; i<PDPageGetNumAnnots(pdPage); i++) { annot = PDPageGetAnnot(pdPage, i); if (PDAnnotIsValid(annot) && PDAnnotGetSubtype(annot) == ASAtomFromString("Link")) { ... } }</pre> |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDAnnotNotifyDidChange

```
void PDAnnotNotifyDidChange (PDAnnot annot, ASAtom key,  
                             ASInt32 err);
```

| | |
|------------------------|--|
| Description | Broadcasts a PDAnnotDidChange notification. Plug-ins must call this method after making any change to a custom annotation. |
| Parameters | <i>anAnnot</i> The annotation that has changed. <i>key</i> The <i>ASAtom</i> corresponding to the name of the key in the annotation's Cos dictionary that is changing. <i>err</i> An error code to pass to any method registered to receive the PDAnnotDidChange notification. Pass zero if the annotation was changed successfully. Pass a nonzero value if an error occurred while changing the annotation. |
| Return Value | None |
| Exceptions | None |
| Notifications | PDAnnotDidChange |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDAnnotNotifyWillChange AVAppRegisterNotification |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDAnnotNotifyWillChange

```
void PDAnnotNotifyWillChange (PDAnnot annot, ASAtom key);
```

| | |
|------------------------|---|
| Description | Broadcasts a PDAnnotWillChange notification. Plug-ins must call this method before making any change to a custom annotation. |
| Parameters | <i>anAnnot</i> The annotation that has changed. <i>key</i> The <i>ASAtom</i> corresponding to the name of the key in the annotation's Cos dictionary that is changing. |
| Return Value | None |
| Exceptions | None |
| Notifications | PDAnnotWillChange |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDAnnotNotifyDidChange AVAppRegisterNotification |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDAnnotSetColor

```
void PDAnnotSetColor (PDAnnot anAnnot,
    const PDColorValue color);
```

| | |
|------------------------|--|
| Description | Sets a note or link annotation's color. Only RGB color specifications are currently supported. |
| Parameters | <p><i>anAnnot</i></p> <p>The note or link annotation whose color is set.</p> <p><i>color</i></p> <p>The annotation's color, which is used as follows:</p> <ul style="list-style-type: none"> • Closed text note—the icon background color • Open, unselected text note—bounding rectangle color • Open, selected text note—color of annotation's title bar • Link annotation—link border color |
| Return Value | None |
| Exceptions | None |
| Notifications | PDAnnotWillChange PDAnnotDidChange |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDAnnotGetColor PDAnnotGetDate PDAnnotGetFlags PDAnnotGetRect PDAnnotGetTitle |
| Example | <pre>for (i=0; i<PDPageGetNumAnnots(pdPage); i++) { annot = PDPageGetAnnot(pdPage, i); if (PDAnnotIsValid(annot) && PDAnnotGetSubtype(annot) == ASAtomFromString("Text")) { if (!PDAnnotGetColor(annot, oldColor)) PDAnnotSetColor(annot, newColor); } }</pre> |

Availability

Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDAnnotSetDate

```
void PDAnnotSetDate (PDAnnot anAnnot, const ATimeRecP date);
```

| | |
|------------------------|---|
| Description | Sets an annotation's date. |
| Parameters | <p><i>anAnnot</i></p> <p>The annotation whose date is set.</p> <p><i>date</i></p> <p>The annotation's time and date.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | PDAnnotWillChange PDAnnotDidChange |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDAnnotGetDate PDAnnotGetColor PDAnnotGetFlags PDAnnotGetRect PDAnnotGetTitle |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDAnnotSetFlags

```
void PDAnnotSetFlags ( PDAnnot anAnnot, ASUns32 flags);
```

Description Sets an annotation's flags.

Parameters *anAnnot*
The annotation whose flags are set.
flags
An OR of the [PDAnnot Flags](#) values.

Return Value None

Exceptions None

Notifications [PDAnnotWillChange](#)
[PDAnnotDidChange](#)

Header File *PDCalls.h*

Related Methods [PDAnnotGetFlags](#)
[PDAnnotGetColor](#)
[PDAnnotGetDate](#)
[PDAnnotGetRect](#)
[PDAnnotGetTitle](#)

Example

```
if (!PDAnnotGetFlags(annot) &
    pdAnnotInvisible)
    PDAnnotSetFlags(annot, pdAnnotInvisible);
```

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDAnnotSetRect

```
void PDAnnotSetRect (PDAnnot anAnnot,
    const ASFixedRect* newBox);
```

Description Sets the size and location of an annotation on its page.

Parameters

anAnnot

The annotation whose location and size are set.

newBox

Pointer to a rectangle that specifies the annotation's bounding rectangle, specified in user space coordinates.

Return Value None

Exceptions None

Notifications [PDAnnotWillChange](#)
[PDAnnotDidChange](#)

Header File *PDCalls.h*

Related Methods [PDAnnotGetRect](#)
[PDAnnotGetColor](#)
[PDAnnotGetDate](#)
[PDAnnotGetFlags](#)
[PDAnnotGetTitle](#)

Example

```
ASFixedRect frect;
for(i=0;i<PDPageGetNumAnnots(pdPage);i++){
    annot = PDPageGetAnnot(pdPage, i);
    if(PDAnnotIsValid(annot) &&
        PDAnnotGetSubtype(annot) ==
        ASAtomFromString("Text"))
    {
        PDAnnotGetRect(annot, &frect);
        /* Modify the annot's rect */
        ...
        /* Set the annot's rect */
        PDAnnotSetRect(annot, &frect);
    }
}
```

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDAnnotSetTitle

```
void PDAnnotSetTitle (PDAnnot anAnnot, const char* str,  
    ASInt32 nBytes);
```

Description

Sets an annotation's label text.

Note: For Roman viewers, this text is always stored in the PDFDocEncoding. For non-Roman character set viewers, this text is stored as PDFDocEncoding or Unicode, depending on the file's creator. Files created in a non-Roman environment contain Unicode versions of these strings; in a Roman environment, files contain PDFDocEncoding versions of these strings.

Parameters

anAnnot

The annotation whose label is set.

str

String containing the label to set.

nBytes

Length of label. The first *len* bytes of *str* are used as the label.

Return Value

None

Exceptions

None

Notifications

[PDAnnotWillChange](#)
[PDAnnotDidChange](#)

Header File

PDCalls.h

Related Methods

[PDAnnotGetTitle](#)
[PDAnnotGetColor](#)
[PDAnnotGetDate](#)
[PDAnnotGetRect](#)
[PDAnnotGetFlags](#)

Example

```
for(i=0;i<PDPageGetNumAnnots(pdPage);i++){
    annot = PDPageGetAnnot(pdPage, i);
    if(PDAnnotIsValid(annot) &&
        PDAnnotGetSubtype(annot) ==
        ASAtomFromString("Text"))
    {
        if(!PDAnnotGetTitle(annot, buf,
            sizeof(buf)))
        {
            strcat(buf, "-RD");
            PDAnnotSetTitle(annot, buf,
                sizeof(buf));
        }
    }
}
```

Availability

Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDAnnotHandler

PDGetAnnotHandlerByName

[PDAnnotHandler](#) PDGetAnnotHandlerByName ([ASAtom](#) name) ;

Description Gets the annotation handler that handles the specified annotation type.

Parameters *name*
Name of the requested annotation handler. The character string for the name can be converted to an *ASAtom* using *ASAtomFromString*.

Return Value The annotation handler that services annotations of type name. Returns the default annotation handler if no handler services the specified annotation type.

Exceptions None

Notifications None

Header File *PDCalls.h*

Related Methods [AVAppGetAnnotHandlerByName](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDRegisterAnnotHandler

```
void PDRegisterAnnotHandler ( PDAnnotHandler handler );
```

Description Registers a handler for an annotation subtype, replacing any previous handler that had been registered for that subtype. The annotation handler is not registered if its [PDAnnotHandlerGetTypeProc](#) returns *NULL*.

Parameters *handler*
Pointer to a structure containing the annotation handler's callbacks. This structure must not be freed after this call, but must be retained.

Return Value None

Exceptions None

Notifications None

Header File *PDCalls.h*

Related Methods [AVAppRegisterAnnotHandler](#).

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDBead

PDBeadAcquirePage

[PDPPage](#) PDBeadAcquirePage ([PDBead](#) bead, [PDDoc](#) doc);

| | |
|------------------------|---|
| Description | Acquires the page on which a bead is located. |
| Parameters | <p><i>bead</i></p> <p>The bead whose page is acquired.</p> <p><i>doc</i></p> <p>The document in which <i>bead</i> is located.</p> |
| Return Value | The page on which the bead resides. The acquired page must be freed using PDPPageRelease when it is no longer needed. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDPPageRelease |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBeadDestroy

```
void PDBeadDestroy (PDBead bead);
```

| | |
|------------------------|--|
| Description | Destroys a bead. |
| Parameters | <i>bead</i> The bead to destroy. |
| Return Value | None |
| Exceptions | pdErrBadBead |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBeadNew PDBeadFromCosObj |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDBeadEqual

```
ASBool PDBeadEqual (PDBead bead, PDBead bead2);
```

| | |
|------------------------|--|
| Description | Tests two beads for equality. This method is useful to detect the end of a thread since the last bead in a thread points to the first. |
| Parameters | <i>bead, bead2</i> The beads to compare. |
| Return Value | <i>true</i> if the two beads are identical, <i>false</i> otherwise. Two beads are equal if and only if their Cos objects are equal (see CosObjEqual). |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | CosObjEqual |
| Example | <pre>fbead = bead = PDThreadGetFirstBead(thread); do { conPage = PDBeadAcquirePage(bead,pdDoc); pageNum = PDPageGetNumber(conPage); /* process pageNum */ ... PDPageRelease(conPage); bead = PDBeadGetNext(bead); } while(PDBeadIsValid(bead) && !PDBeadEqual(fbead, bead));</pre> |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

PDBeadFromCosObj

[PDBead](#) PDBeadFromCosObj ([CosObj](#) obj);

| | |
|------------------------|---|
| Description | Gets the <i>PDBead</i> corresponding to a Cos object, after checking the bead's validity. This method does not copy the object, but is instead the logical equivalent of a type cast. |
| Parameters | <p><i>obj</i></p> <p>Dictionary Cos object for the bead whose <i>PDBead</i> is obtained.</p> |
| Return Value | The <i>PDBead</i> object for the bead. |
| Exceptions | Raises pdErrBadBead if the bead is not valid, as determined by PDBeadIsValid . |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBeadGetCosObj |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBeadGetCosObj

```
CosObj PDBeadGetCosObj ( PDBead bead ) ;
```

| | |
|------------------------|--|
| Description | Gets the Cos object corresponding to a bead. This method does not copy the object, but is instead the logical equivalent of a type cast. |
| Parameters | <p><i>bead</i></p> <p>The bead whose Cos object is obtained.</p> |
| Return Value | Dictionary Cos object for the bead. The contents of the dictionary can be enumerated using CosObjEnum . Returns a <i>NULL</i> Cos object if PDBeadIsValid (<i>bead</i>) returns <i>false</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBeadFromCosObj |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBeadGetIndex

```
ASInt32 PDBeadGetIndex ( PDBead bead );
```

| | |
|------------------------|--|
| Description | Gets the index of a bead in its thread. |
| Parameters | <p><i>bead</i></p> <p>The bead whose index is obtained.</p> |
| Return Value | The index of the bead in its thread. The first bead in a thread has an index of zero. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBeadGetNext

[PDBead](#) PDBeadGetNext ([PDBead](#) bead);

| | |
|------------------------|---|
| Description | Gets the next bead in a thread. |
| Parameters | <p><i>bead</i></p> <p>The bead for which the next bead is obtained.</p> |
| Return Value | The next bead, or a <i>NULL</i> Cos object (cast to a <i>PDBead</i> using PDBeadFromCosObj). <i>PDBeadGetNext</i> on the last bead returns the first bead. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBeadGetPrev PDThreadGetFirstBead PDBeadEqual |

Example

```
fbead = bead =
    PDThreadGetFirstBead(thread);
do {
    conPage = PDBeadAcquirePage(bead,pdDoc);
    pageNum = PDPageGetNumber(conPage);

    /* process pageNum */
    ...

    PDPageRelease(conPage);
    bead = PDBeadGetNext(bead);
} while(PDBeadIsValid(bead) &&
        !PDBeadEqual(fbead, bead));
```

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

PDBeadGetPrev

[PDBead](#) PDBeadGetPrev ([PDBead](#) bead);

| | |
|------------------------|---|
| Description | Gets the previous bead in a thread. |
| Parameters | <p><i>bead</i></p> <p>The bead for which the previous bead is obtained.</p> |
| Return Value | The previous bead, or a <i>NULL</i> Cos object (cast to a <i>PDBead</i> using PDBeadFromCosObj) if this is the first bead in the thread. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBeadGetNext PDThreadGetFirstBead PDBeadEqual |

Example

```
fbead = bead =
    PDThreadGetFirstBead(thread);
do {
    conPage = PDBeadAcquirePage(bead,pdDoc);
    pageNum = PDPageGetNumber(conPage);

    /* process pageNum */
    ...

    PDPageRelease(conPage);
    bead = PDBeadGetNext(bead);
} while(PDBeadIsValid(bead) &&
        !PDBeadEqual(fbead, bead));
```

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

PDBeadGetRect

```
void PDBeadGetRect (PDBead bead, ASFixedRectP rectP);
```

| | |
|-----------------------|--|
| Description | Gets a bead's bounding rectangle. |
| Parameters | <p><i>bead</i></p> <p>The bead whose bounding rectangle is obtained.</p> <p><i>rectP</i></p> <p>(Filled by the method) Pointer to a ASFixedRect specifying the bead's bounding rectangle, specified in user space coordinates.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Method | PDBeadSetRect PDBeadGetIndex PDBeadGetNext PDBeadGetPrev PDBeadGetThread |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBeadGetThread

```
PDThread PDBeadGetThread ( PDBead bead );
```

| | |
|------------------------|---|
| Description | Gets the thread containing the specified bead. |
| Parameters | <p><i>bead</i></p> <p>The bead whose thread is obtained.</p> |
| Return Value | The bead's thread, or a <i>NULL</i> Cos object if the bead does not belong to a thread. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBeadGetRect PDBeadGetIndex PDBeadGetNext PDBeadGetPrev |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBeadInsert

```
void PDBeadInsert (PDBead bead, PDBead newNext);
```

| | |
|------------------------|--|
| Description | Inserts a bead after the specified bead. |
| Parameters | <p><i>bead</i></p> <p>The bead after which <i>newNext</i> will be inserted.</p> <p><i>newNext</i></p> <p>The bead to insert.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBeadNew PDThreadSetFirstBead |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDBeadIsValid

```
ASBool PDBeadIsValid (PDBead bead);
```

Description Tests a bead's validity. This is intended only to ensure that the bead has not been deleted, not to ensure that all necessary information is present and valid.

Parameters *bead*
The bead whose validity is tested.

Return Value *true* if the bead is valid, *false* otherwise.

Exceptions None

Notifications None

Header File *PDCalls.h*

Related Methods [PDBeadEqual](#)

Example

```
fbead = bead =
    PDThreadGetFirstBead(thread);
do {
    conPage = PDBeadAcquirePage(bead, pdDoc);
    pageNum = PDPageGetNumber(conPage);

    /* process pageNum */
    ...

    PDPageRelease(conPage);
    bead = PDBeadGetNext(bead);
} while(PDBeadIsValid(bead) &&
        !PDBeadEqual(fbead, bead));
```

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBeadNew

```
PDBead PDBeadNew (PDPage page, const ASFixedRectP destRect);
```

Description Creates a new bead on the specified page. The newly-created bead is not linked to a thread or another bead. Use [PDThreadSetFirstBead](#) to make the bead the first bead in a thread. Use [PDBeadInsert](#) to link it to another bead.

Parameters

page
The page on which the bead is created.

destRect
Pointer to a [ASFixedRect](#) specifying the bead's bounding rectangle, specified in user space coordinates.

Return Value The newly-created bead.

Exceptions None

Notifications None

Header File *PDCalls.h*

Related Methods [PDThreadSetFirstBead](#)
[PDBeadInsert](#)
[PDBeadFromCosObj](#)
[PDBeadDestroy](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDBeadSetPage

```
void PDBeadSetPage (PDBead bead, PDPage newPage);
```

| | |
|------------------------|--|
| Description | Sets the page for a bead. |
| Parameters | <p><i>bead</i></p> <p>The bead whose page is set.</p> <p><i>newPage</i></p> <p>The page on which <i>bead</i> is located.</p> |
| Return Value | None |
| Exceptions | pdErrBadBead |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDBeadSetRect

```
void PDBeadSetRect ( PDBead bead,
    const ASFixedRectP newDestRect );
```

| | |
|------------------------|--|
| Description | Sets a bead's bounding rectangle. |
| Parameters | <p><i>bead</i></p> <p>The bead whose bounding rectangle is set.</p> <p><i>newDestRect</i></p> <p>Pointer to a ASFixedRect specifying the bead's bounding rectangle, specified in user space coordinates.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBeadGetRect PDBeadGetIndex PDBeadGetNext PDBeadGetPrev PDBeadGetThread |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDBookmark

PDBookmarkAddChild

```
void PDBookmarkAddChild (PDBookmark parent,
    PDBookmark aBookmark);
```

Description Adds *aBookmark* as the last child of *parent*, adjusting the tree containing *parent* appropriately. If *parent* previously had no children, it is open after the child is added.

Parameters

parent
The parent of the bookmark being added.

aBookmark
Bookmark that will become the last child of *aBookmark*. *aBookmark* must have been previously unlinked.

Return Value None

Exceptions None

Notifications [PDBookmarkDidChangePosition](#)

Header File *PDCalls.h*

Related Methods [PDBookmarkAddNewSibling](#)
[PDBookmarkAddNewChild](#)
[PDBookmarkAddSubtree](#)
[PDBookmarkAddPrev](#)
[PDBookmarkAddNext](#)
[PDBookmarkUnlink](#)
[PDBookmarkFromCosObj](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDBookmarkAddNewChild

```
PDBookmark PDBookmarkAddNewChild (PDBookmark aBookmark,  
char* initialText);
```

| | |
|------------------------|---|
| Description | Adds a new bookmark to the tree containing <i>aBookmark</i> , as the new last child of <i>aBookmark</i> . If <i>aBookmark</i> previously had no children, it will be open after the child is added. |
| Parameters | <p><i>aBookmark</i></p> <p>The bookmark to which a new last child is added.</p> <p><i>initialText</i></p> <p>The new bookmark's title.</p> |
| Return Value | The newly-created bookmark. |
| Exceptions | None |
| Notifications | PDBookmarkDidChangePosition PDBookmarkWasCreated |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBookmarkAddNewSibling PDBookmarkAddSubtree PDBookmarkAddPrev PDBookmarkAddNext PDBookmarkAddChild PDBookmarkUnlink |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDBookmarkAddNewSibling

```
PDBookmark PDBookmarkAddNewSibling (PDBookmark aBookmark,  
char* initialText);
```

| | |
|------------------------|---|
| Description | Adds a new bookmark to the tree containing a <i>aBookmark</i> , as the new right sibling. |
| Parameters | <p><i>aBookmark</i></p> <p>The bookmark that will be the left sibling of the new bookmark.</p> <p><i>initialText</i></p> <p>The new bookmark's title.</p> |
| Return Value | The newly-created bookmark. |
| Exceptions | None |
| Notifications | PDBookmarkDidChangePosition PDBookmarkWasCreated |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBookmarkAddChild PDBookmarkAddNewChild PDBookmarkAddNext PDBookmarkAddPrev PDBookmarkAddSubtree PDBookmarkUnlink PDBookmarkFromCosObj |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDBookmarkAddNext

```
void PDBookmarkAddNext ( PDBookmark aBookmark,  
    PDBookmark newNext );
```

| | |
|------------------------|---|
| Description | Adds <i>newNext</i> as the new right sibling to <i>aBookmark</i> . |
| Parameters | <p><i>aBookmark</i></p> <p>The bookmark that will receive a new right sibling.</p> <p><i>newNext</i></p> <p>The bookmark to become the new right sibling of <i>aBookmark</i>. <i>newNext</i> must have been previously unlinked.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | PDBookmarkDidChangePosition |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBookmarkAddNewSibling PDBookmarkAddChild PDBookmarkAddSubtree PDBookmarkAddPrev PDBookmarkAddNewChild PDBookmarkUnlink |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDBookmarkAddPrev

```
void PDBookmarkAddPrev (PDBookmark aBookmark,  
    PDBookmark newPrev);
```

Description Adds *newPrev* as the new left sibling to *aBookmark*, adjusting the tree containing *aBookmark* appropriately.

Parameters

aBookmark

Bookmark that will receive a new left sibling *newPrev*.

newPrev

Bookmark to become the new left sibling of *aBookmark*. *newPrev* must have been previously unlinked.

Return Value None

Exceptions None

Notifications [PDBookmarkDidChangePosition](#)

Header File *PDCalls.h*

Related Methods [PDBookmarkAddNewSibling](#)
[PDBookmarkAddNewChild](#)
[PDBookmarkAddSubtree](#)
[PDBookmarkAddNext](#)
[PDBookmarkAddChild](#)
[PDBookmarkUnlink](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDBookmarkAddSubtree

```
void PDBookmarkAddSubtree (PDBookmark aBookmark,  
    PDBookmark source, char* sourceTitle);
```

Description Adds a copy of the bookmark subtree source to *aBookmark*, as a new last child of *aBookmark*. This new item will have the text value *sourceTitle*, will be open, and will have no destination attribute. *source* must have been previously unlinked. If *aBookmark* previously had no children, it will be open after the subtree is added.

Parameters *aBookmark*
The bookmark to which the subtree *source* will be added as a new last child.

source
The bookmark subtree to add.

sourceTitle
The new bookmark's title.

Return Value None

Exceptions None

Notifications [PDBookmarkWillChange](#)
[PDBookmarkDidChange](#)
[PDBookmarkDidChangePosition](#)

Header File *PDCalls.h*

Related Methods [PDBookmarkAddNewSibling](#)
[PDBookmarkAddNewChild](#)
[PDBookmarkAddPrev](#)
[PDBookmarkAddNext](#)
[PDBookmarkAddChild](#)
[PDBookmarkUnlink](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDBookmarkDestroy

```
void PDBookmarkDestroy (PDBookmark aBookmark);
```

| | |
|------------------------|--|
| Description | Removes a bookmark subtree from the bookmark tree containing it. |
| Parameters | <i>aBookmark</i> The root bookmark of the subtree to remove. |
| Return Value | None |
| Exceptions | None |
| Notifications | PDBookmarkWillDestroy PDBookmarkDidDestroy |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBookmarkUnlink |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDBookmarkEqual

```
ASBool PDBookmarkEqual (PDBookmark aBookmark,
    PDBookmark bookmark2);
```

| | |
|------------------------|--|
| Description | Tests whether or not two bookmarks are equal. Two bookmarks are equal if and only if their Cos objects are equal (see CosObjEqual). |
| Parameters | <i>aBookmark, bookmark2</i> The two bookmarks to compare. |
| Return Value | <i>true</i> if the bookmarks are equal, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | CosObjEqual |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkFromCosObj

[PDBookmark](#) PDBookmarkFromCosObj ([CosObj](#) obj);

| | |
|------------------------|---|
| Description | Converts a Cos dictionary object to a bookmark and checks the validity of the bookmark. This method does not copy the object, but is instead the logical equivalent of a type cast. |
| Parameters | <i>obj</i> Dictionary Cos object to convert to a bookmark. |
| Return Value | Bookmark corresponding to the Cos object. |
| Exceptions | Raises pdErrBadBookmark if the bookmark is not valid as determined by PDBookmarkIsValid . |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBeadGetCosObj |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkGetAction

[PDAction](#) PDBookmarkGetAction ([PDBookmark](#) aBookmark) ;

| | |
|------------------------|---|
| Description | Gets a bookmark's action. After you obtain the action, you can execute it with AVDocPerformAction . |
| Parameters | <p><i>aBookmark</i></p> <p>The bookmark whose action is obtained.</p> <p><i>action</i></p> <p>(Filled by the method) The bookmark's action.</p> |
| Return Value | The bookmark's action. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | AVDocPerformAction PDBookmarkSetAction |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkGetByTitle

```
PDBookmark PDBookmarkGetByTitle (PDBookmark aBookmark,  
char* aName, ASInt32 nameLen, ASInt32 maxdepth);
```

Description

Gets the first bookmark whose title is *aName*.

Note: For Roman viewers, this text is always stored in the PDFDocEncoding. For non-Roman character set viewers, this text is stored as PDFDocEncoding or Unicode, depending on the file's creator. Files created in a non-Roman environment contain Unicode versions of these strings; in a Roman environment, files contain PDFDocEncoding versions of these strings.

Parameters

aBookmark

The root of the bookmark subtree to search.

aName

The text value to search for. Character codes in *aName* are interpreted using the PDFDocEncoding.

nameLen

The length of *aName*.

maxDepth

The number of subtree levels to search, not counting the root level.

- 0 — Only look at *aBookmark*, not at any of its children.
- 1 — Check *aBookmark* and its children, but not any grandchildren or great grandchildren, and so on.
- -1 — Check the entire subtree.

Return Value

The bookmark with the specified title, or a *NULL* Cos object if there is no such bookmark.

Exceptions

None

Notifications

None

Header File

PDCalls.h

Related Methods

None

Availability

Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkGetCosObj

[CosObj](#) PDBookmarkGetCosObj ([PDBookmark](#) aBookmark) ;

| | |
|------------------------|--|
| Description | Obtains the Cos object for a bookmark. This method does not copy the object, but is instead the logical equivalent of a type cast. |
| Parameters | <i>aBookmark</i> The bookmark whose Cos object is obtained. |
| Return Value | Dictionary Cos object for the bookmark. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBookmarkFromCosObj CosObjEnum |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkGetCount

```
ASInt32 PDBookmarkGetCount ( PDBookmark aBookmark );
```

| | |
|------------------------|--|
| Description | Gets the number of open bookmarks in a subtree. |
| Parameters | <p><i>aBookmark</i></p> <p>The root of a subtree to count.</p> |
| Return Value | Number of open boookmarks in the subtree (not including <i>aBookmark</i>). |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkGetFirstChild

[PDBookmark](#) PDBookmarkGetFirstChild ([PDBookmark](#) aBookmark);

| | |
|------------------------|---|
| Description | Gets a bookmark's first child. |
| Parameters | <i>aBookmark</i> The bookmark whose first child is obtained. |
| Return Value | First child of <i>aBookmark</i> , or a <i>NULL</i> Cos object if <i>aBookmark</i> has no children. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBookmarkGetParent PDBookmarkGetLastChild PDBeadGetNext PDBookmarkGetPrev |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkGetIndent

```
ASInt32 PDBookmarkGetIndent ( PDBookmark aBookmark );
```

| | |
|------------------------|--|
| Description | Returns the indentation level of a bookmark in its containing tree. |
| Parameters | <p><i>aBookmark</i></p> <p>The bookmark whose indentation level is obtained.</p> |
| Return Value | The indentation level of <i>aBookmark</i> in its containing tree. The root level has an indentation level of zero. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkGetLastChild

[PDBookmark](#) PDBookmarkGetLastChild ([PDBookmark](#) aBookmark);

| | |
|------------------------|--|
| Description | Gets a bookmark's last child. |
| Parameters | <i>aBookmark</i> The bookmark whose last child is obtained. |
| Return Value | Last child of <i>aBookmark</i> , or a <i>NULL</i> Cos object if <i>aBookmark</i> has no children. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBookmarkGetParent PDBookmarkGetFirstChild PDBookmarkGetNext PDBookmarkGetPrev |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkGetNext

[PDBookmark](#) PDBookmarkGetNext ([PDBookmark](#) aBookmark) ;

| | |
|------------------------|---|
| Description | Gets bookmark's next (right) sibling. |
| Parameters | <p><i>aBookmark</i></p> <p>Bookmark whose right sibling is being obtained.</p> |
| Return Value | <i>aBookmark</i> 's next (right) sibling. Returns a <i>NULL</i> Cos object if <i>aBookmark</i> has no next sibling (that is, it is its parent's last child). |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBookmarkGetParent PDBookmarkGetFirstChild PDBookmarkGetLastChild PDBookmarkGetPrev |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkGetParent

[PDBookmark](#) PDBookmarkGetParent ([PDBookmark](#) aBookmark) ;

| | |
|------------------------|---|
| Description | Gets a bookmark's parent bookmark. |
| Parameters | <i>aBookmark</i> Bookmark whose parent is obtained. |
| Return Value | Parent bookmark of <i>aBookmark</i> , or a <i>NULL</i> Cos object if <i>aBookmark</i> is the root of its tree. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBookmarkGetFirstChild PDBookmarkGetLastChild PDBookmarkGetNext PDBookmarkGetPrev |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkGetPrev

[PDBookmark](#) PDBookmarkGetPrev ([PDBookmark](#) aBookmark) ;

| | |
|------------------------|---|
| Description | Returns bookmark's previous (left) sibling. |
| Parameters | <p><i>aBookmark</i></p> <p>The bookmark whose left sibling is obtained.</p> |
| Return Value | Previous (left) sibling of <i>aBookmark</i> , or a <i>NULL</i> Cos object if bookmark has no previous sibling (it is its parent's first child). |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBookmarkGetParent PDBookmarkGetFirstChild PDBookmarkGetLastChild PDBookmarkGetNext |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkGetTitle

```
ASInt32 PDBookmarkGetTitle (PDBookmark aBookmark, char* buffer,
    ASInt32 bufSize);
```

Description

Gets a bookmark's title.

Note: For Roman viewers, this title text is always stored in the PDFDocEncoding. For non-Roman character set viewers, this text is stored as PDFDocEncoding or Unicode, depending on the file's creator. Files created in a non-Roman environment contain Unicode versions of these strings; in a Roman environment, files contain PDFDocEncoding versions of these strings.

Parameters

aBookmark

Bookmark whose title is obtained.

buffer

(Filled by the method) Buffer into which the title will be written. If *buffer* is non-NULL, it is assumed to be of size *nBytes* + 1. The returned text remains valid only until next PDMModel method call. The text is returned using PDFDocEncoding, and may be converted to a platform's native encoding using [PDXlateToHost](#) or [PDXlateToHostEx](#).

bufSize

Number of bytes to copy, not counting trailing NULL.

Return Value

The number of characters copied into *buffer*.

Exceptions

None

Notifications

None

Header File

PDCalls.h

Related Methods

[PDBookmarkSetTitle](#)
[PDXlateToHost](#)
[PDXlateToHostEx](#)

Availability

Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkHasChildren

```
ASBool PDBookmarkHasChildren ( PDBookmark aBookmark );
```

| | |
|------------------------|--|
| Description | Tests whether or not a bookmark has children. |
| Parameters | <i>aBookmark</i> The bookmark to test. |
| Return Value | <i>true</i> if <i>aBookmark</i> has any children, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkIsOpen

```
ASBool PDBookmarkIsOpen (PDBookmark aBookmark);
```

| | |
|------------------------|--|
| Description | Tests whether or not a bookmark is open. An open bookmark shows all its children. |
| Parameters | <i>aBookmark</i> The bookmark to test. |
| Return Value | <i>true</i> if the bookmark is open, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBookmarkSetOpen |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkIsValid

```
ASBool PDBookmarkIsValid (PDBookmark aBookmark);
```

Description Tests whether or not a bookmark is valid. This is intended only to ensure that the bookmark has not been deleted, not to ensure that all necessary information is present and valid.

Parameters *aBookmark*
The bookmark whose validity is being tested.

Return Value *true* if the bookmark is valid, *false* otherwise.

Exceptions None

Notifications None

Header File *PDCalls.h*

Related Methods [PDBookmarkEqual](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkSetAction

```
void PDBookmarkSetAction (PDBookmark aBookmark,  
    PDAction action);
```

| | |
|------------------------|--|
| Description | Sets a bookmark's action. |
| Parameters | <p><i>aBookmark</i></p> <p>The bookmark whose action is set.</p> <p><i>action</i></p> <p>The bookmark's action.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | PDBookmarkWillChange PDBookmarkDidChange PDNameTreeNameRemoved |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBookmarkGetAction |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDBookmarkSetOpen

```
void PDBookmarkSetOpen (PDBookmark aBookmark, ASBool isOpen);
```

Description Opens or closes a bookmark. An open bookmark shows its children, while a closed bookmark does not.

Parameters *aBookmark*
The bookmark to open or close.
isOpen
true if the bookmark is opened, *false* if the bookmark is closed.

Return Value None

Exceptions None

Notifications [PDBookmarkWillChange](#)
[PDBookmarkDidChange](#)

Header File *PDCalls.h*

Related Methods [PDBookmarkIsOpen](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDBookmarkSetTitle

```
void PDBookmarkSetTitle (PDBookmark aBookmark, const char* str,
    ASInt32 nBytes);
```

Description

Sets a bookmark's title.

Note: For Roman viewers, this title text is always stored in the PDFDocEncoding. For non-Roman character set viewers, this text is stored as PDFDocEncoding or Unicode, depending on the file's creator. Files created in a non-Roman environment contain Unicode versions of these strings; in a Roman environment, files contain PDFDocEncoding versions of these strings.

Parameters

aBookmark

Bookmark whose title is set.

str

Read-only string containing the bookmark's new title. The text must be encoded using PDFDocEncoding. Strings encoded using a platform's native encoding can be converted to PDFDocEncoding using [PDXlateToPDFDocEnc](#) or [PDXlateToPDFDocEncEx](#).

nBytes

Number of bytes of *str* to copy.

Return Value

None

Exceptions

Raises [pdErrBookmarksError](#) if there is an error setting the title.

Notifications

[PDBookmarkWillChange](#)
[PDBookmarkDidChange](#)

Header File

PDCalls.h

Related Methods

[PDBookmarkGetTitle](#)

Availability

Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDBookmarkUnlink

```
void PDBookmarkUnlink (PDBookmark aBookmark);
```

| | |
|------------------------|--|
| Description | Unlinks a bookmark from the bookmark tree that contains it, and adjusts the tree appropriately. |
| Parameters | <i>aBookmark</i> The bookmark to unlink. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDBookmarkDestroy |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDCharProc

PDCharProcEnum

```
void PDCharProcEnum (PDCharProc charProc,
    PDGraphicEnumMonitor mon, void* clientData);
```

Description Enumerates the graphic description of a single character procedure for a Type 3 font. To enumerate all the character procedures in a Type 3 font (but not their graphic descriptions), use [PDFontEnumCharProcs](#).

Parameters *charProc*
The character procedure whose graphic description is enumerated.

mon
Pointer to a structure containing user-supplied callbacks that are called for each drawing operator on a page. Enumeration halts if any procedure returns *false*.

clientData
Pointer to user-supplied data to pass to each monitor routine that is called.

Return Value None

Exceptions None

Notifications None

Header File *PDCalls.h*

Related Methods [PDFontEnumCharProcs](#)
[PDDocEnumFonts](#)
[PDDocEnumLoadedFonts](#)
[PDCharProcGetCosObj](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | M, W, U | M, W, U |

PDCharProcGetCosObj

```
CosObj PDCharProcGetCosObj (PDCharProc obj);
```

| | |
|------------------------|---|
| Description | Get the stream Cos object associated with the <i>PDCharProc</i> . This method does not copy the object, but is instead the logical equivalent of a type cast. |
| Parameters | <p><i>obj</i></p> <p>The character procedure whose Cos object is obtained.</p> |
| Return Value | The character procedure's stream Cos object. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDFontEnumCharProcs PDCharProcEnum |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDDoc

PDDocAcquire

```
void PDDocAcquire (PDDoc doc);
```

Description Increments a document's reference count. The document will not be closed until the reference count is zero, or the application terminates.

Parameters *doc*
The document whose reference count is incremented.

Return Value None

Exceptions None

Notifications None

Header File *PDCalls.h*

Related Methods [PDDocRelease](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|---------|
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDDocAcquirePage

[PDPage](#) PDDocAcquirePage ([PDDoc](#) doc, ASInt32 pageNum);

| | |
|------------------------|---|
| Description | Obtains a <i>PDPage</i> from a document. Increments the page's reference count. After you are done using the page, release it using PDPageRelease . |
| Parameters | <p><i>doc</i></p> <p>The document containing the page to acquire.</p> <p><i>pageNum</i></p> <p>The page number of the page to acquire. The first page is 0.</p> |
| Return Value | The acquired page. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDPageRelease |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDDocAddThread

```
void PDDocAddThread (PDDoc doc, ASInt32 addAfterIndex,
    PDThread thread);
```

| | |
|------------------------|--|
| Description | Adds an article thread to a document after the specified thread index. |
| Parameters | <p><i>doc</i></p> <p>The document in which the thread is added. Must match that used in the call to PDThreadNew that created the thread.</p> <p><i>addAfterIndex</i></p> <p>The index of the thread after which <i>thread</i> is added.</p> <p><i>thread</i></p> <p>The thread to add.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | PDDocDidAddThread |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDThreadNew PDThreadFromCosObj PDDocRemovePageLabel |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDDocAuthorize

```
PDPerms PDDocAuthorize (PDDoc pdDoc, PDPerms permsWanted,
void* authData);
```

| | |
|------------------------|---|
| Description | Adds permissions to the specified document, if permitted. Calls the PDCryptAuthorizeProc callback of the document's security handler to determine which of the specified permissions will actually be granted. After calling this method, the document's permissions will be the OR of the previous permissions and the permissions granted by the PDCryptAuthorizeProc callback. |
| Parameters | <p><i>pdDoc</i></p> <p>The document for which new permissions are requested.</p> <p><i>permsWanted</i></p> <p>The new permissions being requested. Must be an OR of the PDPerms values.</p> <p><i>authData</i></p> <p>Pointer to data to pass to the PDCryptAuthorizeProc callback of the document's security handler. For the Acrobat viewer's built-in security handler, <i>authData</i> is a <i>char*</i> containing the password.</p> |
| Return Value | The OR of the previous value of the document's permissions field, and the permissions granted by the PDCryptAuthorizeProc callback of the document's security handler. The result will be an OR of the PDPerms values. |
| Exceptions | <p>Raises pdErrNeedCryptHandler if no security handler is associated with <i>pdDoc</i>.</p> <p>Raises whatever exceptions are raised by the security handler's PDCryptAuthorizeProc callback.</p> |
| Notifications | None |
| Header File | <i>PICrypt.h</i> |
| Related Methods | PDDocGetNewCryptHandler PDRegisterCryptHandler |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |
| Available in: | |

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDDocClearFlags

```
void PDDocClearFlags (PDDoc doc, ASInt32 flags);
```

Description Clears flags associated with a document. This method is most frequently used to mark a modified document as clean (by clearing the *PDDocNeedsSave* flag) to avoid bringing up the Save dialog when the file is closed.

Parameters

doc
The document whose flags are cleared.

flags
The flags to clear. Must be an OR of the [PDDocFlags](#) values.

Return Value None

Exceptions None

Notifications None

Header File *PDCalls.h*

Related Methods [PDDocSave](#)
[PDDocClose](#)
[PDDocGetFlags](#)
[PDDocSetFlags](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020001 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDDocClose

```
void PDDocClose (PDDoc doc);
```

Description Closes a document and releases its resources. If *doc* is *NULL*, does nothing. Changes are not saved. You must use [PDDocSave](#) to save any modifications before calling *PDDocClose*.

If the document has been modified but you wish to mark it as clean, use [PDDocClearFlags](#).

Parameters *doc*
The document to close.

Return Value None

Exceptions Raises [pdErrUnableToCloseDueToRefs](#) if there are any outstanding references to objects in the document, and the document will still be valid (its resources will not be released).

Raises [genErrBadUnlock](#) if the document's open count is less than one.

Notifications None

Header File *PDCalls.h*

Related Methods [PDDocSave](#)
[PDDocOpen](#)
[PDDocCreate](#)
[PDDocClearFlags](#)
[PDDocSetFlags](#)

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDDocCreate

[PDDoc](#) PDDocCreate (void);

Description

Creates a new document. The only Cos object in the document will be a Catalog. See Section 6.2 in the [Portable Document Format Reference Manual](#). After the document is created, at least one page must be added using [PDDocCreatePage](#) or [PDDocInsertPages](#) before the Acrobat viewer can display or save the document.

When you are done with the document, you must call [PDDocClose](#) to release the resources used by the *PDDoc*; do *not* call [PDDocRelease](#).

Parameters

None

Return Value

The newly-created document.

Exceptions

None

Notifications

None

Header File

PDCalls.h

Related Methods

[PDDocClose](#)
[PDDocOpen](#)
[PDDocSave](#)
[PDDocCreatePage](#)
[PDDocInsertPages](#)

Availability

Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | — | — |

PDDocCreateNameTree

```
PDNameTree PDDocCreateNameTree ( PDDoc thePDDoc,  
    ASAtom theTree );
```

- Description** Retrieves the name tree inside the Names dictionary with the specified key name, or creates it if it does not exist.
- Parameters**
- thePDDoc*
The document in which the name tree is created.
- theTree*
The name of the name tree to create. A string can be converted to an *ASAtom* using [ASAtomFromString](#).
- Return Value** The newly-created *PDNameTree* for the *PDDoc*. Returns a *NULL PDNameTree* if *pdDoc* has no root dictionary. The return value should be tested with [PDNameTreesValid](#).
- Exceptions** None
- Notifications** None
- Header File** *PDCalls.h*
- Related Methods** [PDNameTreesValid](#)
[PDDocGetNameTree](#)
[PDDocRemoveNameTree](#)
- Example**
- ```
PDNameTree pdNameTree;
PDDoc thePDDoc;

pdNameTree = PDDocCreateNameTree (thePDDoc,
 ASAtomFromString ("docNameTree"));
```
- Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

**Available in:**

| Library API       | 1.0 | 4.0     |
|-------------------|-----|---------|
| Adobe PDF Library | —   | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0  |
|-------------|-----------|------|
| Acrobat     | —         | M, W |
| Reader      | —         | —    |



## PDDocCreatePage

```
PDPage PDDocCreatePage (PDDoc doc, ASInt32 afterPageNum,
ASFixedRect mediaBox);
```

**Description** Creates and acquires a new page. The page is inserted into the document at the specified location. Call [PDPageRelease](#) when you are done using the page.

**Parameters**

*doc*  
The document in which to create a page.

*afterPageNum*  
The page number after which the new page is inserted. The first page is 0. Use *PDBeforeFirstPage* (see *PDExpT.h*) to insert the new page at the beginning of a document.

*mediaBox*  
Rectangle specifying the page's media box, specified in user space coordinates.

**Return Value** The newly-created page.

**Exceptions** None

**Notifications** [PDDocWillExportAnnots](#)  
[PDDocDidExportAnnots](#)  
[PDDocDidChangePages](#)  
[PDDocWillPrintPages](#)  
[PDDocDidChangeThumbs](#)

**Header File** *PDCalls.h*

**Related Methods** [PDPageRelease](#)  
[PDDocDeletePages](#)  
[PDDocImportNotes](#)  
[PDDocReplacePages](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0  |
|-------------|-----------|------|
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDDocCreateStructTreeRoot

```
void PDDocCreateStructTreeRoot (PDDoc pdDoc,
 PDSTreeRoot* treeRoot);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                         |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Creates a new <i>StructTreeRoot</i> element.<br><br>If <i>PDDocCreateStructTreeRoot</i> is called on a <i>PDDoc</i> that already has a structure tree root, it returns without modifying the document.                                                                                                                                                                                                  |
| <b>Parameters</b>      | <p><i>pdDoc</i></p> <p>The <i>PDDoc</i> for which <i>StructTreeRoot</i> element is created.</p> <p><i>treeRoot</i></p> <p>(Filled by the method) The newly-created <i>StructTreeRoot</i> element.</p>                                                                                                                                                                                                   |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>Exceptions</b>      | Raises an exception if <i>pdDoc</i> already has a <i>StructTreeRoot</i> .                                                                                                                                                                                                                                                                                                                               |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>Header File</b>     | <i>PDSWriteCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Related Methods</b> | <a href="#">PDDocGetStructTreeRoot</a><br><a href="#">PDSTreeRootGetRoleMap</a><br><a href="#">PDSTreeRootGetClassMap</a><br><a href="#">PDSTreeRootGetElementFromID</a><br><a href="#">PDDocRemoveStructTreeRoot</a><br><a href="#">PDSTreeRootCreateRoleMap</a><br><a href="#">PDSTreeRootRemoveRoleMap</a><br><a href="#">PDSTreeRootCreateClassMap</a><br><a href="#">PDSTreeRootRemoveClassMap</a> |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                                                                                                                              |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Reader      | —         | M, W, U |

## PDDocCreateTextSelect

```
PDTextSelect PDDocCreateTextSelect (PDDoc doc, ASInt32 pageNum,
ASFixedRect* boundingRect);
```

### Description

Creates a text selection that includes all words totally or partially enclosed by a rectangle. The text selection can then be set as the current selection using [AVDocSetSelection](#).

*Note: When this method is used to create a text selection on a rotated page, you must pass in a rotated boundingRect.*

### Parameters

*doc*

The document in which a text selection is created.

*pageNum*

The page number on which the text selection is created.

*boundingRect*

Pointer to a rectangle specifying the text selection's bounding rectangle, specified in user space coordinates.

### Return Value

The newly-created text selection.

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDTextSelectDestroy](#)  
[AVDocSetSelection](#)  
[PDTextSelectEnumQuads](#)  
[PDTextSelectEnumText](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocCreateThumbs

```
void PDDocCreateThumbs (PDDoc doc, ASInt32 firstPage,
 ASInt32 lastPage, PDThumbCreationServer server,
 void* serverClientData, ASAAtom colorSpace,
 ASInt32 bitsPerComponent, ASInt32 hiVal, char* lookupTable,
 ProgressMonitor progMon, void* progMonClientData,
 CancelProc cancelProc, void* cancelProcClientData);
```

### Description

Creates thumbnail images for the specified range of pages. Thumbnail images are only created for pages that have none.

Use as large of a page range as possible because the color space object is shared by all the thumbnails created by a single invocation of this method (that is, if you call this method separately for each page, there will be duplicate color space objects).

See Section 7.11 in the [Portable Document Format Reference Manual](#) for more details on color spaces.

*Note: In version 2.0 and 2.1 of the Acrobat viewers, firstPage and lastPage are ignored, and thumbnail images are always created for all pages.*

See the Windows chapter of Technical Note #5167, [Acrobat API Development](#), for additional important information about creating thumbnails on the Windows platform.

### Parameters

*doc*

The document for which thumbnail images are created.

*firstPage*

The page number of the first page for which thumbnails are created. The first page is 0.

*lastPage*

The page number of the last page for which thumbnails are created. The constant *PDLastPage* (see *PDExpT.h*) can also be used.

*server*

A server (set of callback procedures) that provides the sampled image used as the thumbnail image. Pass *NULL* to use the default server.

## *serverClientData*

User-supplied data to pass to the thumbnail creation server.

## *colorSpace*

The color space in which the thumbnail data is represented. It must be *DeviceRGB*.

Thumbnails may be created in either a direct or an indexed color space; however, it is strongly recommended that you use indexed color spaces over direct color spaces. Using direct color spaces with this version of Acrobat may cause bad looking thumbnails.

To specify a direct color space, pass 0 for *hiVal* and *NULL* for *lookupTable*. To specify an indexed color space, pass the appropriate values in *hiVal* and *lookupTable*.

Direct color spaces in Windows are supported in Acrobat 3.0. Prior to Acrobat 3.0 in Windows, you had to use indexed color spaces.

## *bitsPerComponent*

The number of bits per color component in the thumbnail image's data. 8 is the only valid value.

## *hiVal*

Used only for indexed color space; pass 0 for direct color spaces, as described in *colorSpace*.

*hiVal* specifies the highest valid index in *lookupTable*. Because indices start at 0, the number of entries in *lookupTable* is *hiVal* + 1. *hiVal* must be 0 for device color spaces.

## *lookupTable*

Used only for indexed color space; pass *NULL* for direct color spaces, as described in *colorSpace*.

*lookupTable* is a table that maps data values to colors. Used only for indexed color spaces. Must be *NULL* for device color spaces.

For an indexed color space, the size of the lookup table must be  $(hiVal + 1) \times sizeof(ASUns8) \times 3$ , where the 3 arises because an RGB color space has three color components.



## *progMon*

A monitor to call to display thumbnail creation progress. Use [AVAppGetDocProgressMonitor](#) to obtain the standard progress monitor to pass for this parameter. *NULL* may be passed, in which case no progress monitor is used.

## *progMonClientData*

User-supplied data to pass to *progMon* each time it is called. Should be *NULL* if *progMon* is *NULL*.

## *cancelProc*

A procedure to call frequently to allow the user to cancel thumbnail creation. Use [AVAppGetCancelProc](#) to obtain a cancel proc for this parameter. May be *NULL*, in which case no cancel proc is used.

## *cancelProcClientData*

User-supplied data to pass to *cancelProc* each time it is called. Should be *NULL* if *cancelProc* is *NULL*.

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Return Value</b>    | None                                                                                                     |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | <a href="#">PDDocDidChangeThumbs</a>                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDDocDeleteThumbs</a>                                                                        |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher. |

## Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | —         | —       |

## PDDocCreateWordFinder

```
PDWordFinder PDDocCreateWordFinder (PDDoc doc,
ASUnsl6* outEncInfo, char** outEncVec, char** ligatureTbl,
ASInt16 algVersion, ASUnsl6 rdFlags, void* clientData);
```

### Description

Creates a word finder that is used to extract text in the host encoding from a PDF file. The word finder may either be used by [PDWordFinderEnumWords](#) (which enumerates words one-by-one) or by [PDWordFinderAcquireWordList](#) (which fills a table with all the words on a page).

*Note: The word finder also extracts text from Form XObjects that are executed in the page contents. For information about Form XObjects, see Section 7.12.7 in the [Portable Document Format Reference Manual](#).*

A default ligature table is used, containing the following ligatures: fi, ff, fl, ffi, ffl, ch, cl, ct, ll, ss, fs, st, oe, ae, OE, AE. The glyph name is substituted for the ligature.

This method also works for non-Roman (CJK or Chinese-Japanese-Korean) viewers. In this case, words are extracted to the host encoding. Users desiring Unicode output must use [PDDocCreateWordFinderUCS](#), which does the extraction for Roman or non-Roman text.

The type of [PDWordFinder](#) determines the encoding of the string returned by [PDWordGetString](#). For instance, if [PDDocCreateWordFinderUCS](#) is used to create the word finder, [PDWordGetString](#) returns only Unicode.

For CJK viewers, words are stored internally using CID encoding. For more information on CIDFonts and related topics, see Section 7.7.9 in the [Portable Document Format Reference Manual](#). For detailed information on CIDFonts, see Technical Note #5092, *CID-Keyed Font Technology Overview*, and Technical Note #5014, *Adobe CMap and CIDFont Files Specification*.

### Parameters

*doc*

The document on which the word finder is used.

## *outEncInfo*

Array of 256 flags, specifying the type of character at each position in the encoding. Each flag is an OR of the [Character Type Codes](#). If *outEncInfo* is *NULL*, the platform's default encoding info is used. Use *outEncInfo* and *outEncVec* together; for every *outEncInfo* use a corresponding *outEncVec* to specify the character at that position in the encoding.

Regardless of the characters specified in *outEncInfo* as word separators, a default set of word separators is used (see [Glyph Names of Word Separators](#)). There is no way to change the list of characters that are considered to be word separators.

## *outEncVec*

Array of 256 *NULL*-terminated strings that are the glyph names in encoding order. See the discussion of character names in Section 5.3 of the *PostScript Language Reference Manual, Second Edition*. If *outEncVec* is *NULL*, the platform's default encoding vector is used. For non-UNIX Roman systems, it is MacRomanEncoding in Mac OS and WinAnsiEncoding in Windows. In UNIX (except HP-UX) Roman systems, it is ISO8859-1 (ISO Latin-1); for HP-UX, it is HP-ROMAN8. See Appendix C in the [Portable Document Format Reference Manual](#) for descriptions of MacRomanEncoding, WinAnsiEncoding, and PDFDocEncoding.

*Note: Use this parameter with outEncInfo. See outEncInfo for more information.*

## *ligatureTbl*

A *NULL*-terminated array of *NULL*-terminated strings. Each string is the glyph name of a ligature in the font. When a word contains a ligature, the glyph name of the ligature is substituted for the ligature (for example, ff is substituted for the ff ligature). This table must be terminated with *NULL*.

If *ligatureTbl* is *NULL*, a default ligature table is used, containing the following ligatures: fi, ff, fl, ffi, ffl, ch, cl, ct, ll, ss, fs, st, oe, ae, OE, AE.

## *algVersion*

The version of the word-finding algorithm to use. If *WF\_LATEST\_VERSION* (see *PDExpT.h*), the most recent version is used. Pass 0 if your plug-in doesn't care.

## *rdFlags*

Word-finding options that determine the tables filled when using [PDWordFinderAcquireWordList](#). Must be an OR of one or more of the [Word Finder Sort Order Flags](#).

## *clientData*

Pointer to user-supplied data to pass to the newly-created word finder.

|                        |                                                                                                                                                                                                                                                        |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Return Value</b>    | The newly-created word finder.                                                                                                                                                                                                                         |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                   |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                   |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                       |
| <b>Related Methods</b> | <a href="#">PDDocCreateWordFinderUCS</a><br><a href="#">PDWordFinderEnumWords</a><br><a href="#">PDWordFinderAcquireWordList</a><br><a href="#">PDWordFinderDestroy</a><br><a href="#">PDWordGetCharacterTypes</a><br><a href="#">PDWordFilterWord</a> |

## Example

```
ACCB1 ASBool ACCB2 WordCounterProc (
 PDWordFinder WordFinder, PDWord word,
 ASInt32 PageNum, void* fileNum)
{
 ASUns32 cur;

 cur = (ASUns32) PDWordGetCharOffset(word);
 WordCnt += (cur - gLast);
 gLast = cur;
 return true;
}

WordFinder = PDDocCreateWordFinder(
 CurrentPDDoc, NULL, NULL, NULL, 0,
 WXE_PDF_ORDER, NULL);
PDWordFinderEnumWords(WordFinder, PageNum,
 ASCallbackCreateProto(PDWordProc,
 &WordCounterProc), NULL);
PDWordFinderDestroy(WordFinder);
```

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocCreateWordFinderUCS

```
PDWordFinder PDDocCreateWordFinderUCS (PDDoc doc,
ASInt16 algVersion, ASUns16 rdFlags, void* clientData);
```

### Description

Creates a word finder that is used to extract text in Unicode format from a PDF file. The word finder may either be used by [PDWordFinderEnumWords](#) (which enumerates words one-by-one) or by [PDWordFinderAcquireWordList](#) (which fills a table with all the words on a page).

*Note: The word finder also extracts text from Form XObjects that are executed in the page contents. For information about Form XObjects, see Section 7.12.7 in the [Portable Document Format Reference Manual](#).*

*Note: PDDocCreateWordFinderUCS is useful for converting non-Roman text (CJK or Chinese-Japanese-Korean) to Unicode. This method also converts Roman text to Unicode in any document.*

[PDDocCreateWordFinder](#) also works for non-Roman character set viewers. For [PDDocCreateWordFinder](#), words are extracted to the host encoding. Users desiring Unicode output should use [PDDocCreateWordFinderUCS](#).

The type of [PDWordFinder](#) determines the encoding of the string returned by [PDWordGetString](#). If [PDDocCreateWordFinderUCS](#) is used to create the word finder, [PDWordGetString](#) returns only Unicode.

*Note: There is no way to detect Unicode strings returned by [PDWordGetString](#), since there is no UCS header (FEFF) added to each string returned.*

In CJK viewers, words are stored internally using CID encoding. For more information on CIDFonts and related topics, see Section 7.7.9 in the [Portable Document Format Reference Manual](#). For detailed information on CIDFonts, see Technical Note #5092, *CID-Keyed Font Technology Overview*, and Technical Note #5014, *Adobe CMap and CIDFont Files Specification*.

## Parameters

*doc*

The document on which the word finder is used.

*algVersion*

The version of the word-finding algorithm to use. If *WF\_LATEST\_VERSION* (see *PDExpT.h*), the most recent version is used. Set to 0 if your plug-in doesn't care.

*rdFlags*

Word-finding options that determine the tables filled when using [PDWordFinderAcquireWordList](#). Must be an OR of one or more of the [Word Finder Sort Order Flags](#).

*clientData*

Pointer to user-supplied data to pass to the newly-created word finder.

## Return Value

The newly-created word finder.

## Exceptions

None

## Notifications

None

## Header File

*PDCalls.h*

## Related Methods

[PDDocCreateWordFinder](#)  
[PDWordFinderEnumWords](#)  
[PDWordFinderAcquireWordList](#)  
[PDWordFinderDestroy](#)  
[PDWordGetCharacterTypes](#)  
[PDWordFilterWord](#)

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

## Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |





## PDDocDeletePages

```
void PDDocDeletePages (PDDoc doc, ASInt32 firstPage,
 ASInt32 lastPage, ProgressMonitor progMon,
 void* progMonClientData);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Deletes the specified pages, inclusively.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document from which pages are deleted.</p> <p><i>firstPage</i></p> <p>The page number of the first page to delete. The first page is 0.</p> <p><i>lastPage</i></p> <p>The page number of the last page to delete.</p> <p><i>progMon</i></p> <p>A progress monitor. Use <a href="#">AVAppGetDocProgressMonitor</a> to obtain the default progress monitor. <i>NULL</i> may be passed, in which case no progress monitor is used.</p> <p><i>progMonClientData</i></p> <p>Pointer to user-supplied data passed to <i>mon</i> each time it is called. Should be <i>NULL</i> if <i>progMon</i> is <i>NULL</i>.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| <b>Notifications</b>   | <a href="#">PDDocWillChangePages</a><br><a href="#">PDDocWillDeletePages</a><br><a href="#">PDDocDidDeletePages</a><br><a href="#">PDDocDidChangePages</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>Related Methods</b> | <a href="#">PDDocImportNotes</a><br><a href="#">PDDocReplacePages</a><br><a href="#">PDDocMovePage</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>Available in:</b>   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDDocDeleteThumbs

```
void PDDocDeleteThumbs (PDDoc doc, ASInt32 firstPage,
 ASInt32 lastPage, ProgressMonitor progMon,
 void* progMonClientData);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | <p>Deletes thumbnail images for a range of pages in a document.</p> <p>In version 2.0 of the Acrobat viewers, <i>firstPage</i> and <i>lastPage</i> are ignored, and thumbnail images are always deleted from all pages.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document from which thumbnail images are deleted.</p> <p><i>firstPage</i></p> <p>The page number of the first page in <i>doc</i> whose thumbnail image is deleted. The first page is 0.</p> <p><i>lastPage</i></p> <p>The page number of the last page in <i>doc</i> whose thumbnail image is deleted.</p> <p><i>progMon</i></p> <p>A monitor to call to display thumbnail deletion progress. Use <a href="#">AVAppGetDocProgressMonitor</a> to obtain the standard progress monitor to pass for this parameter. <i>NULL</i> may be passed, in which case no progress monitor is used.</p> <p><i>progMonClientData</i></p> <p>Pointer to user-supplied data to pass to <i>progMon</i>. Should be <i>NULL</i> if <i>progMon</i> is <i>NULL</i>.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>Notifications</b>   | <a href="#">PDDocDidChangeThumbs</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>Related Methods</b> | <a href="#">PDDocCreateThumbs</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>Available in:</b>   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDDocEnumFonts

```
void PDDocEnumFonts (PDDoc doc, ASInt32 firstPage,
 ASInt32 lastPage, PDFontEnumProc proc, void* clientData,
 ProgressMonitor progMon, void* progMonClientData);
```

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>   | Enumerates all the fonts in the specified page range. This may take a considerable amount of time for a large page range.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>Parameters</b>    | <p><i>doc</i></p> <p>The document whose fonts are enumerated.</p> <p><i>firstPage</i></p> <p>The page number of the first page for which fonts are enumerated. The first page is 0.</p> <p><i>lastPage</i></p> <p>The page number of the last page for which fonts are enumerated.</p> <p><i>proc</i></p> <p>User-supplied callback to call for each font. Enumeration terminates if <i>proc</i> returns <i>false</i>.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>proc</i> each time it is called.</p> <p><i>progMon</i></p> <p>A progress monitor. Use <a href="#">AVAppGetDocProgressMonitor</a> to obtain the standard progress monitor. <i>NULL</i> may be passed, in which case no progress monitor is used.</p> <p><i>progMonClientData</i></p> <p>Pointer to user-supplied data to pass to <i>progMon</i> each time it is called. Should be <i>NULL</i> if <i>progMon</i> is <i>NULL</i>.</p> |
| <b>Return Value</b>  | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Exceptions</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Notifications</b> | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Header File</b>   | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

**Related Methods**

- [PDDocEnumLoadedFonts](#)
- [PDFontGetCosObj](#)
- [PDFontAcquireEncodingArray](#)
- [PDFontAcquireXlateTable](#)
- [PDFontEnumCharProcs](#)
- [PDFontGetEncodingIndex](#)
- [PDFontGetFontMatrix](#)
- [PDFontGetBBox](#)
- [PDFontGetCharSet](#)
- [PDFontGetMetrics](#)
- [PDFontGetName](#)
- [PDFontGetSubtype](#)
- [PDFontGetWidths](#)
- [PDFontIsEmbedded](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocEnumLoadedFonts

```
void PDDocEnumLoadedFonts (PDDoc doc, PDFontEnumProc proc,
 void* clientData);
```

**Description** Enumerates all the fonts that have been encountered so far. A font is loaded when a page that uses it is processed. This typically happens when a page is drawn or its thumbnail image is created.

**Parameters**

*doc*  
The document whose loaded fonts are enumerated.

*proc*  
User-supplied callback to call for each loaded font. Enumeration terminates if *proc* returns *false*.

*clientData*  
Pointer to user-supplied data to pass to *proc* each time it is called.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocEnumFonts](#)  
[PDFontEnumCharProcs](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDDocEnumResources

```
void PDDocEnumResources (PDDoc pdDoc, ASInt32 startPage,
 ASInt32 endPage, ASAtom resourceType,
 CosObjEnumProc enumProc, void* clientData);
```

### Description

Enumerates the specified type of page resources, for a specified range of pages.

This method enumerates resources in each page's Resources dictionary (ColorSpace resources, Fonts, ExtGState objects, or others). In addition, it looks inside inline images and page contents to enumerate ColorSpace resources that are not in the **Resources** dictionary, such as *DeviceGray*, *DeviceRGB*, and *DeviceCMYK*.

### Parameters

*pdDoc*

The document whose resources are enumerated.

*startPage*

The first page whose resources are enumerated. (The first page in a document is 0.)

*endPage*

The last page whose resources are enumerated.

*resourceType*

Resource type to enumerate. Must be one of the valid PDF resource types, such as Font, ColorSpace, XObject, Pattern, and so on, described in Section 7.5 in the [Portable Document Format Reference Manual](#).

Pass *ASAtomNull* to enumerate all resource types.

*enumProc*

User-supplied callback to call once for each resource of the specified type. The resource is presented as a *CosObj*, and it is the first parameter of *enumProc* (the second parameter is unused).

*clientData*

User-supplied data to pass to *enumProc* each time it is called.

### Return Value

None



**Exceptions** [genErrBadParm](#)

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocEnumFonts](#)  
[PDEEnumElements](#)  
[PDELogDump](#)  
[PDEObjectDump](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |     |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat     | —         | —   |
| Reader      | —         | —   |

## PDDocExportNotes

```
CosDoc PDDocExportNotes (PDDoc sourceDoc, ASFileSys fileSys,
ASPathName path, void* progMon, void* monClientData,
PDDocWillExportAnnotCallback exportFilter, ASInt32* numNotesP);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Creates a document containing empty pages plus text annotations (notes) from <i>sourceDoc</i> . Does not create a new document if <i>sourceDoc</i> contains no notes.                                                                                                                                                                                                                                                                           |
| <b>Parameters</b>      | <p><i>sourceDoc</i></p> <p>The document from which notes are exported.</p> <p><i>fileSys</i></p> <p>Currently unused.</p> <p><i>path</i></p> <p>Currently unused.</p> <p><i>progMon</i></p> <p>Currently unused.</p> <p><i>monClientData</i></p> <p>Currently unused.</p> <p><i>exportFilter</i></p> <p>User-supplied routine that selects which notes to export.</p> <p><i>numNotesP</i></p> <p>If non-NULL, the number of notes exported.</p> |
| <b>Return Value</b>    | The <i>CosDoc</i> of the document created to hold the exported notes.                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>Notifications</b>   | <a href="#">PDDocDidExportAnnots</a>                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Related Methods</b> | <a href="#">PDDocImportCosDocNotes</a><br><a href="#">PDDocImportNotes</a>                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                                                                                                                                                                        |
| <b>Available in:</b>   |                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## PDDocFindPageNumForLabel

```
ASInt32 PDDocFindPageNumForLabel (PDDoc pdDoc,
 const char* labelStr, ASInt32 labelLen);
```

**Description** Finds the first page in the document with the specified label.

**Parameters** *pdDoc*  
The document to search for the page named in *labelStr*.

*labelStr*  
Label of the page to find.

*labelLen*  
Length of *labelStr*.

**Return Value** The page index, or -1 if no such page exists.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocGetLabelForPageNum](#)  
[PDDocGetPageLabel](#)  
[PDDocRemovePageLabel](#)  
[PDDocSetPageLabel](#)  
[PDPageLabelEqual](#)  
[PDPageLabelFromCosObj](#)  
[PDPageLabelGetCosObj](#)  
[PDPageLabelGetPrefix](#)  
[PDPageLabelGetStart](#)  
[PDPageLabelGetStyle](#)  
[PDPageLabellsValid](#)  
[PDPageLabelNew](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

**Available in:**

| Library API       | 1.0 | 4.0     |
|-------------------|-----|---------|
| Adobe PDF Library | —   | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDDocFromCosDoc

[PDDoc](#) PDDocFromCosDoc ([CosDoc](#) cosDoc);

|                        |                                                                                                                                                                                                    |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Obtains the <i>PDDoc</i> associated with a <i>CosDoc</i> .                                                                                                                                         |
| <b>Parameters</b>      | <p><i>cosDoc</i></p> <p>The document for which a <i>PDDoc</i> is needed. This is the Cos level object that represents the PDF.</p>                                                                 |
| <b>Return Value</b>    | The <i>PDDoc</i> associated with <i>cosDoc</i> .                                                                                                                                                   |
| <b>Exceptions</b>      | <p>Raises <a href="#">genErrBadParm</a> if the <i>CosDoc</i> is not valid.</p> <p>Raises <a href="#">pdErrNoPDDocForCosDoc</a> if there is no <i>PDDoc</i> associated with this <i>cosDoc</i>.</p> |
| <b>Notifications</b>   | None                                                                                                                                                                                               |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                   |
| <b>Related Methods</b> | <a href="#">AVDocGetPDDoc</a><br><a href="#">PDPageGetDoc</a><br><a href="#">PDDocGetCosDoc</a>                                                                                                    |
| <b>Example</b>         | <pre>CosDoc cosDoc; PDDoc pdDoc;  pdDoc = PDDocFromCosDoc (cosDoc);</pre>                                                                                                                          |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                           |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | —         | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | —         | M, W    |
| Reader            | —         | M, W, U |

## PDDocGetBookmarkRoot

[PDBookmark](#) PDDocGetBookmarkRoot ( [PDDoc](#) pdDoc ) ;

|                        |                                                                                                                                                                                                   |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the root of the document's bookmark tree. The return value is valid even if document's bookmark tree is empty (that is, even if there is no <i>Outlines</i> key in the underlying PDF file). |
| <b>Parameters</b>      | <i>pdDoc</i><br>Document whose root bookmark is obtained.                                                                                                                                         |
| <b>Return Value</b>    | The document's root bookmark.                                                                                                                                                                     |
| <b>Exceptions</b>      | None                                                                                                                                                                                              |
| <b>Notifications</b>   | None                                                                                                                                                                                              |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                  |
| <b>Related Methods</b> | None                                                                                                                                                                                              |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                          |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocGetCosDoc

[CosDoc](#) PDDocGetCosDoc ( [PDDoc](#) doc );

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets a document's <i>CosDoc</i> .                                                                        |
| <b>Parameters</b>      | <i>doc</i><br>The document whose <i>CosDoc</i> is obtained.                                              |
| <b>Return Value</b>    | The document's <i>CosDoc</i> .                                                                           |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">CosObjGetDoc</a>                                                                             |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher. |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDDocGetCryptHandlerClientData

```
void* PDDocGetCryptHandlerClientData (PDDoc doc);
```

|                        |                                                                                                                                                                                  |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the client data for the encryption handler associated with the <i>PDDoc</i> . This is the client data provided as a parameter in <a href="#">PDRegisterCryptHandlerEx</a> . |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The <i>PDDoc</i> whose encryption handler client data is obtained.</p>                                                                                      |
| <b>Return Value</b>    | Client data for the encryption handler associated with the <i>PDDoc</i> . Returns <i>NULL</i> if there is no encryption handler or no client data was provided.                  |
| <b>Exceptions</b>      | None                                                                                                                                                                             |
| <b>Notifications</b>   | None                                                                                                                                                                             |
| <b>Header File</b>     | <i>PICrypt.h</i>                                                                                                                                                                 |
| <b>Related Methods</b> | <a href="#">PDRegisterCryptHandlerEx</a>                                                                                                                                         |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher.                                                                         |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocGetFile

```
ASFile PDDocGetFile (PDDoc doc);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the <i>ASfile</i> for a document.                                                                   |
| <b>Parameters</b>      | <i>doc</i><br>The document whose <i>ASFile</i> is obtained.                                              |
| <b>Return Value</b>    | The document's <i>ASFile</i> .                                                                           |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | None                                                                                                     |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher. |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocGetFlags

```
ASInt32 PDDocGetFlags (PDDoc doc);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets information about the document's file and its state.                                                |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document whose flags are obtained.</p>                                          |
| <b>Return Value</b>    | Flags field, containing an OR of the <a href="#">PDDocFlags</a> values.                                  |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDDocGetFlags</a><br><a href="#">PDDocClearFlags</a>                                         |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher. |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocGetFullScreen

```
ASBool PDDocGetFullScreen (PDDoc pdDoc);
```

|                        |                                                                                                                                                                                   |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Indicates whether or not the document will open in full-screen mode. This provides an alternative to calling <a href="#">PDDocGetPageLabel</a> to test for <i>PDFFullScreen</i> . |
| <b>Parameters</b>      | <i>pdDoc</i><br>The document to test.                                                                                                                                             |
| <b>Return Value</b>    | <i>true</i> if the <i>PDDoc</i> is in full-screen mode, <i>false</i> otherwise.                                                                                                   |
| <b>Exceptions</b>      | None                                                                                                                                                                              |
| <b>Notifications</b>   | None                                                                                                                                                                              |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                  |
| <b>Related Methods</b> | <a href="#">PDDocSetFullScreen</a>                                                                                                                                                |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher.                                                                          |

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocGetID

```
ASInt32 PDDocGetID (PDDoc doc, ASInt32 nElemNum,
 ASUns8* buffer, ASInt32 bufferSize);
```

**Description** Gets a *PDDocId* array element. See Section 6.12 in the [Portable Document Format Reference Manual](#) for a description of file IDs.

**Parameters**

*doc*  
The document whose file ID is obtained.

*nElemNum*  
The element number to get from the document's file ID. Must be one of the following:  
0 — The permanent ID  
1 — The changing ID

*buffer*  
(Filled by the method) If buffer is non-NULL, then up to the *bufferLen* bytes of the ID will be written to the buffer.

*bufferSize*  
Length of *buffer*, in bytes.

**Return Value** The number of bytes in the ID.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** None

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Reader      | M, W, U   | M, W, U |

## PDDocGetInfo

```
ASInt32 PDDocGetInfo (PDDoc doc, const char* infoKey,
 char* buffer, ASInt32 bufSize);
```

### Description

Gets the value of a key in a document's Info dictionary. See Section 6.10 on info dictionaries in the [Portable Document Format Reference Manual](#) for information about Info dictionaries. All values in the Info dictionary should be strings; other data types such as numbers and booleans should not be used as values in the Info dictionary.

Users may define their own Info dictionary entries. In this case, it is strongly recommended that the key have the developer's prefix assigned by the Adobe Developers Association.

*Note: For Roman viewers, this text is always stored in the PDFDocEncoding. For non-Roman character set viewers, this text is stored as PDFDocEncoding or Unicode, depending on the file's creator. Files created in a non-Roman environment contain Unicode versions of these strings; in a Roman environment, files contain PDFDocEncoding versions of these strings.*

### Parameters

*doc*

The document whose Info dictionary key is being obtained.

*infoKey*

The name of the Info dictionary key whose value is obtained.

*buffer*

*(Filled by the method)* Buffer containing the value associated with *infoKey*. If *buffer* is *NULL*, the method will just return the number of bytes required.

*bufSize*

The maximum number of bytes that can be written into *buffer*.

**Return Value** If *buffer* is *NULL*, the number of bytes in the specified key's value. If *buffer* is not *NULL*, returns the number of bytes copied into *buffer*, excluding the terminating *NULL*. You must pass at least the *length* + 1 as the buffer size since the routine adds a '\0' terminator to the data, even though the data is not a C string (it can contain embedded '\0's).

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocSetInfo](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDDocGetLabelForPageNum

```
ASInt32 PDDocGetLabelForPageNum (PDDoc pdDoc, ASInt32 pageNum,
 char* buffer, ASInt32 bufferLen);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Retrieves the label string associated with the given page number. The page number is returned in host encoding and is truncated to the length of the buffer.                                                                                                                                                                                                                                                                                                                  |
| <b>Parameters</b>      | <p><i>pdDoc</i></p> <p>Document containing the page for which a label is requested.</p> <p><i>pageNum</i></p> <p>The number of the page whose label is requested.</p> <p><i>buffer</i></p> <p>If a label exists for <i>pageNum</i>, it will be placed in this buffer.</p> <p><i>bufferLen</i></p> <p>Length of the label (<i>NULL</i> terminated).</p>                                                                                                                        |
| <b>Return Value</b>    | The length of the resulting label. If no such page number exists, the resulting string will be the ASCII representation of <i>pageNum</i> + 1.                                                                                                                                                                                                                                                                                                                                |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Related Methods</b> | <a href="#">PDDocFindPageNumForLabel</a><br><a href="#">PDDocGetPageLabel</a><br><a href="#">PDDocRemovePageLabel</a><br><a href="#">PDDocSetPageLabel</a><br><a href="#">PDPageLabelEqual</a><br><a href="#">PDPageLabelFromCosObj</a><br><a href="#">PDPageLabelGetCosObj</a><br><a href="#">PDPageLabelGetPrefix</a><br><a href="#">PDPageLabelGetStart</a><br><a href="#">PDPageLabelGetStyle</a><br><a href="#">PDPageLabellsValid</a><br><a href="#">PDPageLabelNew</a> |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                                                                                                                                                                                                      |

**Available in:**

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDDocGetNameTree

```
PDNameTree PDDocGetNameTree (PDDoc thePDDoc, ASAtom theTree);
```

**Description** Retrieves a name tree, with the key name specified in *theTree*, from the Names dictionary of *thePDDoc*.

**Parameters** *thePDDoc*  
The document containing the name tree desired.  
*theTree*

The name of the tree requested. This can be created by passing a string to the [ASAtomFromString](#) method.

**Return Value** The *PDNameTree* requested or *NULL* if the *PDNameTree* does not exist.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [ASAtomFromString](#)  
[PDNameTreeIsValid](#)  
[PDDocCreateNameTree](#)  
[PDDocRemoveNameTree](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDDocGetNewCryptHandler

[ASAtom](#) PDDocGetNewCryptHandler ([PDDoc](#) doc);

**Description** Gets the specified document's new security handler (that is, the security handler that will be used after the document is saved).  
If the document does not have a new security handler, returns the document's current security handler.

**Parameters** *doc*  
The document whose new security handler is obtained.

**Return Value** The *ASAtom* corresponding to the *pdfName* of the document's new security handler. Returns *ASAtomNull* if the document does not have a new security handler.

**Exceptions** None

**Notifications** None

**Header File** *PICrypt.h*

**Related Methods** [PDDocSetNewCryptHandler](#)  
[PDRegisterCryptHandler](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocGetNewSecurityData

```
void* PDDocGetNewSecurityData (PDDoc doc);
```

|                        |                                                                                                                                                                                                              |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the security data structure for the specified document's new security handler. Use <a href="#">PDDocGetSecurityData</a> to get the security data structure for the document's current security handler. |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document whose new security data structure is obtained.</p>                                                                                                                         |
| <b>Return Value</b>    | The security data structure for the document's new security handler.                                                                                                                                         |
| <b>Exceptions</b>      | None                                                                                                                                                                                                         |
| <b>Notifications</b>   | None                                                                                                                                                                                                         |
| <b>Header File</b>     | <i>PICrypt.h</i>                                                                                                                                                                                             |
| <b>Related Methods</b> | <a href="#">PDDocGetSecurityData</a><br><a href="#">PDDocNewSecurityData</a><br><a href="#">PDDocSetNewCryptHandler</a><br><a href="#">PDDocSetNewSecurityData</a>                                           |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                     |

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocGetNewSecurityInfo

```
void PDDocGetNewSecurityInfo (PDDoc pdDoc, ASUns32* secInfo);
```

**Description** Gets the security information from the specified document's new security handler. Calls the [PDCryptGetSecurityInfoProc](#) callback of the document's new security handler. No permissions are required to call this method.

**Parameters**

*pdDoc*

The document whose new security information is obtained.

*secInfo*

(Filled by the method) The document's new security information.

The value mst be an OR of the [Security Info Flags](#). Set to **pdInfoCanPrint** | **pdInfoCanEdit** | **pdInfoCanCopy** | **pdInfoCanEditNotes** (see [PDPerms](#)) if the document's new security handler does not have a [PDCryptGetSecurityInfoProc](#) callback.

**Return Value** None

**Exceptions** Raises only those exceptions raised by the new security handler's [PDCryptGetSecurityInfoProc](#) callback.

**Notifications** None

**Header File** *PICrypt.h*

**Related Methods** [PDRegisterCryptHandler](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |



## PDDocGetNumPages

```
ASInt32 PDDocGetNumPages (PDDoc doc);
```

|                        |                                                                                                                                                                     |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the number of pages in a document.                                                                                                                             |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document for which the number of pages is obtained.</p>                                                                                    |
| <b>Return Value</b>    | The number of pages in the document. Remember to subtract one from this value if you are going to pass it to a PD-level method that takes a zero-based page number. |
| <b>Exceptions</b>      | None                                                                                                                                                                |
| <b>Notifications</b>   | None                                                                                                                                                                |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                    |
| <b>Related Methods</b> | None                                                                                                                                                                |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                            |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDDocGetNumThreads

```
ASInt32 PDDocGetNumThreads (PDDoc doc);
```

**Description** Gets the number of article threads in a document.

**Parameters** *doc*  
The document whose article thread count is obtained.

**Return Value** The number of article threads in the document.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** None

**Example**

```
size = PDDocGetNumThreads(pdDoc);
if (size)
{
 for(i = 0;i<size;i++)
 {
 /* get the right thread */
 thread = PDDocGetThread(pdDoc, i);
 threadIndex = i;
 len = PDThreadGetInfo(thread, "Title",
 msg, sizeof(msg));
 if(!strcmp(msg, "Contents"))
 break;
 }
}
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDDocGetOpenAction

[PDAction](#) PDDocGetOpenAction ([PDDoc](#) doc);

**Description** Gets the value of the *OpenAction* key in the Catalog dictionary, which is the action performed when the document is opened. After you obtain the action, you can execute it with [AVDocPerformAction](#).

**Parameters** *doc*  
The document whose open action is obtained.

**Return Value** The document's open action. Invalid if there is no *OpenAction* key in the Catalog dictionary (this can be tested with [PDViewDestIsValid](#)).

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [AVDocPerformAction](#)  
[PDDocSetOpenAction](#)  
[PDActionFromCosObj](#)  
[PDActionNew](#)  
[PDActionNewFromDest](#)  
[PDActionNewFromFileSpec](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDDocGetPageLabel

PDPageLabel PDDocGetPageLabel (PDDoc pdDoc, ASInt32 pageNum,  
ASInt32\* firstPage, ASInt32\* lastPage);

|                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>   | Returns the label that is in effect for the given page.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>Parameters</b>    | <p><i>pdDoc</i></p> <p>The document for which a page label is desired.</p> <p><i>pageNum</i></p> <p>The page number of the page for which a label is requested.</p> <p><i>firstPage</i></p> <p>If non-<i>NULL</i>, the number of the page that the page label is attached to.</p> <p><i>lastPage</i></p> <p>If non-<i>NULL</i>, the number of the next higher page that has a label object attached to it. If there is no such object, the number of pages in the document.</p> <p>Setting <i>lastPage</i> to non-<i>NULL</i> forces the implementation to perform another search of the page label tree, with some slight performance impact.</p> |
| <b>Return Value</b>  | The label that is in effect for the given page. If there is no label object in effect, this method returns an invalid page label object and <i>firstPage</i> and <i>lastPage</i> will be set to -1.                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Exceptions</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>Notifications</b> | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>Header File</b>   | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

**Related Methods**    [PDDocFindPageNumForLabel](#)  
[PDDocGetLabelForPageNum](#)  
[PDDocRemovePageLabel](#)  
[PDDocSetPageLabel](#)  
[PDPageLabelEqual](#)  
[PDPageLabelFromCosObj](#)  
[PDPageLabelGetCosObj](#)  
[PDPageLabelGetPrefix](#)  
[PDPageLabelGetStart](#)  
[PDPageLabelGetStyle](#)  
[PDPageLabelIsValid](#)  
[PDPageLabelNew](#)

**Availability**        Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

**Available in:**

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDDocGetPageMode

[PDPageMode](#) PDDocGetPageMode ( [PDDoc](#) doc );

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the value of the <i>PageMode</i> key in the Catalog dictionary.                                     |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document whose page mode is obtained.</p>                                       |
| <b>Return Value</b>    | Page mode value from PDF <i>Catalog</i> dictionary.                                                      |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDDocSetPageLabel</a><br><a href="#">AVDocSetViewMode</a>                                    |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher. |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocGetPermissions

```
PDPerms PDDocGetPermissions (PDDoc doc);
```

|                        |                                                                                                                |
|------------------------|----------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the permissions for the specified document. You can set permissions with <a href="#">PDDocAuthorize</a> . |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document whose permissions are obtained.</p>                                          |
| <b>Return Value</b>    | A bit field indicating the document's permissions. It is an OR of the <a href="#">PDPerms</a> values.          |
| <b>Exceptions</b>      | None                                                                                                           |
| <b>Notifications</b>   | None                                                                                                           |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                               |
| <b>Related Methods</b> | <a href="#">PDDocAuthorize</a>                                                                                 |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.       |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocGetSecurityData

```
void* PDDocGetSecurityData (PDDoc doc);
```

|                        |                                                                                                                                                                                                        |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the security data structure for the specified document's current security handler. Use <a href="#">PDDocGetNewSecurityData</a> to get the data structure for the document's new security handler. |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document whose security data structure is obtained.</p>                                                                                                                       |
| <b>Return Value</b>    | The document's current security data structure.                                                                                                                                                        |
| <b>Exceptions</b>      | None                                                                                                                                                                                                   |
| <b>Notifications</b>   | None                                                                                                                                                                                                   |
| <b>Header File</b>     | <i>PICrypt.h</i>                                                                                                                                                                                       |
| <b>Related Methods</b> | <a href="#">PDDocGetNewSecurityData</a>                                                                                                                                                                |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                               |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDDocGetStructTreeRoot

```
ASBool PDDocGetStructTreeRoot (PDDoc pdDoc,
 PDSTreeRoot* treeRoot);
```

|                        |                                                                                                                                |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the structure tree root for a document.                                                                                   |
| <b>Parameters</b>      | <i>pdDoc</i><br>The <i>PDDoc</i> whose root is obtained.<br><i>treeRoot</i><br>(Filled by the method) The structure tree root. |
| <b>Return Value</b>    | <i>true</i> if structure tree root found, <i>false</i> otherwise.                                                              |
| <b>Exceptions</b>      | None                                                                                                                           |
| <b>Notifications</b>   | None                                                                                                                           |
| <b>Header File</b>     | <i>PDSReadCalls.h</i>                                                                                                          |
| <b>Related Methods</b> | <a href="#">PDDocCreateStructTreeRoot</a><br><a href="#">PDDocRemoveStructTreeRoot</a>                                         |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                      |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDDocGetThread

[PDThread](#) PDDocGetThread ([PDDoc](#) doc, ASInt32 index);

**Description** Gets an article thread having the specified index.

**Parameters** *doc*  
The document containing the article thread.  
*index*  
The index of the article thread to obtain.

**Return Value** The specified article thread.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocAddThread](#)  
[PDDocRemovePageLabel](#)  
[PDThreadIsValid](#)

**Example**

```
size = PDDocGetNumThreads(pdDoc);
if(size)
{
 for(i = 0;i<size;i++)
 {
 /* get the right thread */
 thread = PDDocGetThread(pdDoc, i);
 threadIndex = i;
 len = PDThreadGetInfo(thread, "Title",
 msg, sizeof(msg));
 if(!strcmp(msg, "Contents"))
 break;
 }
}
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocGetThreadIndex

```
ASInt32 PDDocGetThreadIndex (PDDoc doc, PDThread thread);
```

|                        |                                                                                                                              |
|------------------------|------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the index of the specified article thread.                                                                              |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document containing the thread.</p> <p><i>thread</i></p> <p>The thread whose index is obtained.</p> |
| <b>Return Value</b>    | The index of <i>thread</i> in <i>doc</i> . Returns -1 if <i>thread</i> is not in <i>doc</i> .                                |
| <b>Exceptions</b>      | None                                                                                                                         |
| <b>Notifications</b>   | None                                                                                                                         |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                             |
| <b>Related Methods</b> | None                                                                                                                         |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                     |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDDocGetVersion

```
void PDDocGetVersion (PDDoc doc, ASInt16* majorP,
 ASInt16* minorP);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                      |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the major and minor PDF document versions. This is the PDF version of the document, which is specified in the header of a PDF file in the string “%PDF-xx.yy” where xx is the major version and yy is the minor version. For example, version 1.2 has the string “%PDF-1.2”. See Section 5.1 in the <a href="#">Portable Document Format Reference Manual</a> . |
| <b>Parameters</b>      | <i>doc</i><br>The document whose version is obtained.<br><i>majorP</i><br>(Filled by the method) The major version number.<br><i>minorP</i><br>(Filled by the method) The minor version number.                                                                                                                                                                      |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                     |
| <b>Related Methods</b> | None                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                             |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocGetWordFinder

```
PDWordFinder PDDocGetWordFinder (PDDoc docP,
ASInt16 WXEVersion);
```

**Description** Gets the word finder associated with a document. It is not necessary to destroy the word finder returned by this method.

**Parameters** *pdDoc*  
The document whose word finder is obtained.

*WXEVersion*  
The version of the word finder to get.

**Return Value** The document's word finder. Returns *NULL* if the document does not have a word finder or its version does not match the version requested.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocCreateWordFinder](#)  
[PDDocCreateWordFinderUCS](#)

**Example**

```
PDWordFinder WordFinder =
 PDDocGetWordFinder(pdDoc,
 WF_LATEST_VERSION);
if(WordFinder)
{
 PDWordFinderEnumWords(WordFinder,
 PageNum, ASCallbackCreateProto(
 PDWordProc, &WordCounterProc), NULL);
}
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocImportCosDocNotes

```
ASInt32 PDDocImportCosDocNotes (PDDoc doc, CosDoc src,
 const char* noteTitle, ASInt32 noteTitleLen,
 PDColorValue color, void* progMon, void* monClientData,
 PDDocWillImportAnnotCallback importFilter);
```

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>  | Adds text annotations from <i>sourceDoc</i> to <i>doc</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>Parameters</b>   | <p><i>doc</i></p> <p>The document that will receive the imported annotations.</p> <p><i>src</i></p> <p>The document from which the annotations will be imported.</p> <p><i>noteTitle</i></p> <p>Not currently used.</p> <p><i>noteTitleLen</i></p> <p>Not currently used.</p> <p><i>color</i></p> <p>If non-<i>NULL</i>, the color attribute of imported annotations. color indicates the color space (<b>PDDeviceGray</b>, <b>PDDeviceRGB</b>, <b>PDDeviceCMYK</b>), and color values for the annotation.</p> <p><i>progMon</i></p> <p>If supplied, a procedure to call regularly to update a progress bar for the user.</p> <p><i>monClientData</i></p> <p>If supplied, a pointer to private data buffer used by <i>progMon</i>.</p> <p><i>importFilter</i></p> <p>A user-supplied procedure that will be called to provide a filtering process, allowing only desired annotations to import.</p> |
| <b>Return Value</b> | The number of notes imported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>Exceptions</b>   | Raises an exception if the given object has the wrong Cos type. Also raises exceptions if storage is exhausted or file access fails.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |



**Notifications**      [PDDocDidImportAnnots](#)  
[PDDocWillImportAnnots](#)

**Header File**        *PDCalls.h*

**Related Methods**   [PDDocImportNotes](#)  
[PDDocExportNotes](#)

**Availability**        Plug-ins: Available if *PL\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

**Available in:**

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## PDDocImportNotes

```
ASInt32 PDDocImportNotes (PDDoc doc, PDDoc sourceDoc,
 void* progMon, void* monClientData,
 PDDocWillImportAnnotCallback importFilter);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                     |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Adds text annotations (notes) from <i>sourceDoc</i> to <i>doc</i> .                                                                                                                                                                                                                                                                                                                                                 |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document to which the notes are exported to.</p> <p><i>sourceDoc</i></p> <p>The document from which the notes are exported from.</p> <p><i>progMon</i></p> <p>User-supplied progress monitor.</p> <p><i>monClientData</i></p> <p>Data supplied by the monitoring routine.</p> <p><i>importFilter</i></p> <p>User-supplied filter which determines what type of notes will be exported.</p> |
| <b>Return Value</b>    | The number of notes imported.                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>Exceptions</b>      | Raises an exception if the given object has the wrong Cos type. Also raises exceptions if storage is exhausted or file access fails.                                                                                                                                                                                                                                                                                |
| <b>Notifications</b>   | <a href="#">PDDocDidImportAnnots</a><br><a href="#">PDDocWillImportAnnots</a>                                                                                                                                                                                                                                                                                                                                       |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>Related Methods</b> | <a href="#">PDDocExportNotes</a><br><a href="#">PDDocImportCosDocNotes</a>                                                                                                                                                                                                                                                                                                                                          |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                                                                                                                                            |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0  |
|-------------|-----------|------|
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## PDDocInsertPages

```
void PDDocInsertPages (PDDoc doc, ASInt32 mergeAfterThisPage,
 PDDoc doc2, ASInt32 startPage, ASInt32 numPages,
 ASUns16 insertFlags, ProgressMonitor progMon,
 void* progMonClientData, CancelProc cancelProc,
 void* cancelProcClientData);
```

### Description

Inserts *numPages* pages from *doc2* into *doc*. All annotations, and anything else associated with the page (such as a thumbnail image) are copied from the *doc2* pages to the new pages in *doc*. This method will not insert if *doc* equals *doc2*.

### Parameters

*doc*

The document into which pages are inserted. This document must have at least one page.

*mergeAfterThisPage*

The page number in *doc* after which pages from *doc2* are inserted. The first page is 0. If *PDBeforeFirstPage* (see *PDExpT.h*) is used, the pages are inserted before the first page in *doc*. Use *PDLastPage* to insert pages after the last page in *doc*.

*doc2*

The document containing the pages that are inserted into *doc*.

*startPage*

The page number of the first page in *doc2* to insert into *doc*. The first page is 0.

*numPages*

The number of pages in *doc2* to insert into *doc*. Use *PDAAllPages* to insert all pages from *doc2* into *doc*.

## *insertFlags*

Flags that determine what additional information is copied from *doc2* into *doc*. Must be an OR of the following (see *PDExpT.h*):

*PDInsertAll* — Inserts the entire document *doc2* into the document *doc*. This operation not only inserts the pages themselves, but also merges other document data from *doc2* into *doc*. In particular, the following happens:

1. The bookmark tree of *doc2* is merged into the bookmark tree of *doc* by copying it as a new first-level subtree of *doc*'s bookmark tree root, of which it becomes the last child. If *doc* has no bookmark tree, it acquires one identical to the bookmark tree from *doc2*.
2. Named destinations from *doc2* (of PDF 1.1 and later) are copied into *doc*. If there are named destinations in *doc2* with the same name as some named destination in *doc*, the ones in *doc* retain their names and the copied named destinations are given new names based on the old ones with distinguishing digits added. Actions and bookmarks referring to the old names are made to refer to the new names after being copied into *doc*.
3. Document logical structure from *doc2* is copied into *doc*. The top-level children of the structure tree root of *doc2* are copied as new top-level children of the structure tree root of *doc*; a structure tree root is created in *doc* if there was none before. The role maps of the two structure trees are merged, with name conflicts resolved in favor of the role mappings present in *doc*. Attribute objects are not copied, nor is class map information from *doc2* merged into that for *doc*.

If *PDInsertAll* flag is *not* set, pages copied from *doc2* into *doc* will have their structure backpointer information stripped off.

*PDInsertBookmarks* — Inserts bookmarks as well as pages.

*PDInsertThreads* — Inserts threads as well as pages.

## *progMon*

A progress monitor. Use [AVAppGetDocProgressMonitor](#) to obtain the default progress monitor. *NULL* may be passed, in which case no progress monitor is used.

## *progMonClientData*

Pointer to user-supplied data to pass to *progMon* each time it is called. Should be *NULL* if *progMon* is *NULL*.

## *cancelProc*

A cancel procedure. Use *AVAppGetCancelProc* to obtain the current cancel procedure. May be *NULL*, in which case no cancel proc is used.

## *cancelProcClientData*

Pointer to user-supplied data to pass to *cancelProc* each time it is called. Should be *NULL* if *cancelProc* is *NULL*.

## Return Value

None

## Exceptions

Among others:

Raises [pdErrOpNotPermitted](#) unless *doc* is editable and *doc2* is not encrypted or the owner opened it.

Raises [pdErrCantUseNewVersion](#) if *doc2* is a newer major version than the Acrobat viewer understands.

Raises [pdErrTooManyPagesForInsert](#) if the insertion would result in a document with too many pages.

Raises [genErrBadParm](#) if *mergeAfterThisPage* is an invalid page number or *doc* has no pages.

Raises [genErrNoMemory](#) if there is insufficient memory to perform the insertion.

Raises [pdErrWhileRecoverInsertPages](#) if an error occurs while trying to recover from an error during inserting.

## Notifications

[PDDocWillExportAnnots](#)  
[PDDocDidExportAnnots](#)  
[PDDocDidChangePages](#)  
[PDDocWillChangePages](#)  
[PDDocDidChangeThumbs](#)  
[PDDocDidAddThread](#)

## Header File

*PDCalls.h*

**Related Methods**    [AVAppGetCancelProc](#)  
[AVAppGetDocProgressMonitor](#)  
[PDDocCreatePage](#)  
[PDDocDeletePages](#)  
[PDDocMovePage](#)  
[PDDocReplacePages](#)

**Example**

```
/* Extract pages from a document and put
 them in a new document. */
ASFixedRect mediaBoxRect;
PDDoc oldDoc, newDoc = PDDocCreate();

PDDocCreatePage(newDoc, 0, mediaBoxRect);
PDDocInsertPages(newDoc, mergeAfterThisPage,
 oldDoc, start, numPages, flags, NULL,
 NULL, NULL, NULL);
PDDocDeletePages(newDoc, 0, 1, NULL, NULL);
```

**Availability**    Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDDocMovePage

```
void PDDocMovePage (PDDoc doc, ASInt32 moveToAfterThisPage,
 ASInt32 pageToMove);
```

|                        |                                                                                                                                                                                                                                                                                                                                       |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Moves one page in a document.                                                                                                                                                                                                                                                                                                         |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document in which a page is moved.</p> <p><i>moveToAfterThisPage</i></p> <p>The new location of the page to move. The first page is 0. May either be a page number, or the constant <i>PDBeforeFirstPage</i> (see <i>PDExpT.h</i>).</p> <p><i>pageToMove</i></p> <p>The page number of the page to move.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                  |
| <b>Exceptions</b>      | <p>Among others:</p> <p>Raises <a href="#">genErrBadParm</a> if <i>moveAfterThisPage</i> or <i>pageToMove</i> is invalid.</p>                                                                                                                                                                                                         |
| <b>Notifications</b>   | <a href="#">PDDocWillMovePages</a><br><a href="#">PDDocDidMovePages</a><br><a href="#">PDDocDidChangePages</a><br><a href="#">PDDocWillChangePages</a>                                                                                                                                                                                |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                      |
| <b>Related Methods</b> | <a href="#">PDDocImportNotes</a><br><a href="#">PDDocReplacePages</a><br><a href="#">PDDocDeletePages</a>                                                                                                                                                                                                                             |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                              |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | —         | —       |





## PDDocNewSecurityData

```
void* PDDocNewSecurityData (PDDoc doc);
```

### Description

Creates a security data structure appropriate for the specified document's new security handler. The new security handler must have been previously set using [PDDocSetNewCryptHandler](#). The structure is created by calling the new security handler's [PDCryptNewSecurityDataProc](#).

After calling *PDDocNewSecurityData*, fill the structure as appropriate, then call [PDDocSetNewSecurityData](#) with the structure. [PDDocSetNewSecurityData](#) frees the data using [ASfree](#).

### Parameters

*doc*

The document for which a security data structure is created.

### Return Value

The newly-created security data structure.

### Exceptions

Raises [pdErrOpNotPermitted](#) if *pdPermSecure* (see [PDPerms](#)) has not been granted for *doc*.

Raises [pdErrNeedCryptHandler](#) if the document does not have a new security handler.

### Notifications

None

### Header File

*PICrypt.h*

### Related Methods

[PDDocSetNewCryptHandler](#)  
[PDDocSetNewSecurityData](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDDocOpen

```
PDDoc PDDocOpen (ASPathName fileName, ASFileSys fileSys,
PDAuthProc authProc, ASBool doRepair);
```

### Description

Opens the specified document. If the document is already open, returns a reference to the already opened *PDDoc*. You must call [PDDocClose](#) once for every successful open. If the call fails and the exception is [pdErrNeedRebuild](#), then call again with *doRepair* set to *true*. This allows the application to decide whether to perform the time-consuming repair operation.

### Parameters

*fileName*

Pathname to the file, specified in whatever format is correct for *fileSys*.

*fileSys*

Pointer to an [ASFileSysRec](#) containing the file system in which the file resides. If NULL, uses the default file system.

*authProc*

Authorization callback, called only if the file has been secured. This callback should obtain whatever information is needed to determine whether or not the user is authorized to open the file, then call [PDDocAuthorize](#) (which returns the permissions that the authentication data enables).

The Acrobat viewer's built-in authorization procedure requires the user to enter a password, and allows the user to try three times before giving up.

If the *authProc* requires data, use [PDDocOpenEx](#) instead of *PDDocOpen*.

*doRepair*

If *true*, attempt to repair the file if it is damaged. If *false*, do not attempt to repair the file if it is damaged.

### Return Value

The newly-opened document.

## Exceptions

Among others:

Raises [pdErrNeedPassword](#) if the file is encrypted and *authProc* is *NULL* or returns *false*.

Raises [pdErrNotEnoughMemoryToOpenDoc](#) or [genErrNoMemory](#) if there is insufficient memory to open the document.

Raises [pdErrNeedRebuild](#) if the document needs to be rebuilt, and *doRepair* is *false*.

Raises [pdErrBadOutlineObj](#) if the Outlines object appears to be invalid (if the value of the *Outlines* key in the Catalog is not a *NULL* or dictionary object).

Raises [pdErrBadRootObj](#) if the Catalog object (as returned by [CosDocGetObjByID](#)) is not a dictionary.

Raises [pdErrBadBaseObj](#) if the Pages tree appears to be invalid (if the value of the *Pages* key in the Catalog is not a *NULL* or dictionary object).

Raises [pdErrTooManyPagesForOpen](#) if the document contains too many pages.

Raises [cosSynErrNoHeader](#) if the document's header appears to be bad.

Raises [cosSynErrNoStartXRef](#) if no end-of-file line can be located.

Raises [cosErrRebuildFailed](#) if *doRepair* is *true*, and rebuild failed.

## Notifications

None

## Header File

*PDCalls.h*

## Related Methods

[PDDocClose](#)  
[PDDocCreate](#)  
[PDDocAuthorize](#)  
[PDDocOpenEx](#)

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocOpenEx

```
PDDoc PDDocOpenEx (ASPathName fileName, ASFileSys fileSys,
 PDAuthProcEx authProcEx, void* authProcClientData,
 ASBool doRepair);
```

### Description

Opens the specified document. If the document is already open, returns a reference to the already opened *PDDoc*. You must call [PDDocClose](#) once for every successful open. If the call fails and the exception is [pdErrNeedRebuild](#), then call again with *doRepair true*. This allows the application to decide whether to perform the time-consuming repair operation.

### Parameters

*fileName*

Pathname to the file, specified in whatever format is correct for *fileSys*.

*fileSys*

Pointer to an [ASFileSysRec](#) containing the file system in which the file resides.

*authProcEx*

Authorization callback, called only if the file is encrypted. This callback should obtain whatever information is needed to determine whether or not the user is authorized to open the file, then call [PDDocAuthorize](#) (which returns the permissions that the authentication data enables).

The Acrobat viewer's built-in authorization procedure requires the user to enter a password, and allows the user to try three times before giving up.

*authProcClientData*

Pointer to user-supplied data to pass to *authProcEx* each time it is called.

*doRepair*

If *true*, attempt to repair the file if it is damaged. If *false*, do not attempt to repair the file if it is damaged.

### Return Value

The newly-opened document.

## Exceptions

Among others:

Raises [pdErrNeedPassword](#) if the file is encrypted and *authProcEx* is *NULL* or returns *false*.

Raises [pdErrNotEnoughMemoryToOpenDoc](#) or [genErrNoMemory](#) if there is insufficient memory to open the document.

Raises [pdErrNeedRebuild](#) if the document needs to be rebuilt, and *doRepair* is *false*.

Raises [pdErrBadOutlineObj](#) if the Outlines object appears to be invalid (if the value of the *Outlines* key in the Catalog is not a *NULL* or dictionary object).

Raises [pdErrBadRootObj](#) if the Catalog object (as returned by [CosDocGetObjByID](#)) is not a dictionary.

Raises [pdErrBadBaseObj](#) if the Pages tree appears to be invalid (if the value of the *Pages* key in the Catalog is not a *NULL* or dictionary object).

Raises [pdErrTooManyPagesForOpen](#) if the document contains too many pages.

Raises [cosSynErrNoHeader](#) if the document's header appears to be bad.

Raises [cosSynErrNoStartXRef](#) if no end-of-file line can be located.

Raises [cosErrRebuildFailed](#) if *doRepair* is *true*, and rebuild failed.

## Notifications

None

## Header File

*PDCalls.h*

## Related Methods

[PDDocClose](#)  
[PDDocCreate](#)  
[PDDocAuthorize](#)  
[PDDocOpen](#)

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

## Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDDocOpenFromASFile

```
PDDoc PDDocOpenFromASFile (ASFile aFile, PDAuthProc authProc,
ASBool doRepair);
```

### Description

Opens the document specified by the *ASFile*. *aFile* must be a valid *ASFile*. It is the caller's responsibility to dispose of the *ASFile* after calling **PDDocClose**.

This method is useful when the document referenced by the *ASFile* is not on the local machine, and it is being retrieved incrementally using the multiread protocol of an *ASFileSys*. If the bytes required to open a *PDDoc* are not yet available, this method will raise the exception **fileErrBytesNotReady**. The client should call **PDDocOpenFromASFile** until this exception is no longer raised.

### Parameters

*aFile*

The *ASFile* to open. The *ASFile* should be released after the *PDDoc* is closed.

*authProc*

Authorization callback, called only if the file is encrypted. This callback should obtain whatever information is needed to determine whether or not the user is authorized to open the file, then call **PDDocAuthorize** (which returns the permissions that the authentication data enables).

The Acrobat viewer's built-in authorization procedure requires the user to enter a password, and allows the user to try three times before giving up.

*doRepair*

If *true*, attempt to repair the file if it is damaged. If *false*, do not attempt to repair the file if it is damaged.

### Return Value

A valid *PDDoc* if successfully opened.

### Exceptions

Raises **pdErrNeedPassword** if the file is encrypted and *authProc* is *NULL* or returns *false*.

Raises **fileErrBytesNotReady** if the bytes required to open a *PDDoc* are not yet available.

### Notifications

None

**Header File** *PDCalls.h*

**Related Methods** [PDDocClose](#)  
[PDDocOpen](#)

**Availability** Plug-ins: Available if *PL\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocOpenFromASFileEx

```
PDDoc PDDocOpenFromASFileEx (ASFile aFile,
 PDAuthProcEx authProcEx, void* authProcClientData,
 ASBool doRepair);
```

### Description

Opens the document specified by the *ASFile*. *aFile* must be a valid *ASFile*. It is the caller's responsibility to dispose of the *ASFile* after calling **PDDocClose**.

This method is useful when the document referenced by the *ASFile* is not on the local machine, and it is being retrieved incrementally using the multiread protocol of an *ASFileSys*. If the bytes required to open a *PDDoc* are not yet available, this method will raise the exception **fileErrBytesNotReady**. The client should call **PDDocOpenFromASFile** until this exception is no longer raised.

### Parameters

*aFile*

The *ASFile* to open. The *ASFile* should be released after the *PDDoc* is closed.

*authProcEx*

Authorization callback, called only if the file is encrypted. This callback should obtain whatever information is needed to determine whether or not the user is authorized to open the file, then call **PDDocAuthorize** (which returns the permissions that the authentication data enables).

The Acrobat viewer's built-in authorization procedure requires the user to enter a password, and allows the user to try three times before giving up.

*authProcClientData*

Pointer to user-supplied data to pass to *authProcEx* each time it is called.

*doRepair*

If *true*, attempt to repair the file if it is damaged. If *false*, do not attempt to repair the file if it is damaged.

### Return Value

A valid *PDDoc* if successfully opened.

|                        |                                                                                                                                                                                                                                                             |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Exceptions</b>      | <p>Raises <a href="#">pdErrNeedPassword</a> if the file is encrypted and <i>authProc</i> is <i>NULL</i> or returns <i>false</i>.</p> <p>Raises <a href="#">fileErrBytesNotReady</a> if the bytes required to open a <i>PDDoc</i> are not yet available.</p> |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                        |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                            |
| <b>Related Methods</b> | <a href="#">PDDocClose</a><br><a href="#">PDDocOpen</a>                                                                                                                                                                                                     |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher.                                                                                                                                                    |

## Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocOpenWithParams

**PDDoc** PDDocOpenWithParams ( [PDDocOpenParams](#) openParams );

|                        |                                                                                                                                                                                                                                 |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Opens the document specified by the <i>ASFile</i> or <i>ASFileSys/ASPathName</i> . If both are set, the <i>ASFile</i> will be used and the <i>fileSys/pathName</i> is ignored.                                                  |
| <b>Parameters</b>      | <p><i>openParams</i></p> <p>A structure that defines which PDF file is opened. Parameters like: file name, file system, an authorization procedure, and a set of flags that define what permissions the user has on a file.</p> |
| <b>Return Value</b>    | The <i>PDDoc</i> for the PDF document described by the structure passed in <i>openParams</i> .                                                                                                                                  |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                            |
| <b>Notifications</b>   | None                                                                                                                                                                                                                            |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                |
| <b>Related Methods</b> | <a href="#">PDDocOpen</a><br><a href="#">PDDocOpenEx</a><br><a href="#">PDDocOpenFromASFile</a><br><a href="#">PDDocOpenFromASFileEx</a>                                                                                        |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                        |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDDocReadAhead

```
void PDDocReadAhead (PDDoc doc, ASUns32 flags,
 void* clientData);
```

**Description** Used for page-at-a-time downloading and byte-serving Acrobat data. If a document is being viewed over a slow file system, *PDDocReadAhead* issues a byte range request for all the data associated with the flags in *flags*.

**Parameters**

*doc*  
The document being read.

*flags*  
Flags describing type of data to read ahead. Must be OR of flags in [PDDocReadAhead Flags](#).

*clientData*  
Currently unused.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** None

**Example**

```
PDDocReadAhead(pdDoc,
 kPDDocReadAheadAcroForms, (void *) NULL);
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDDocReadAheadPages

```
void PDDocReadAheadPages (PDDoc doc, ASInt32 startPage,
 ASInt32 nPages);
```

|                        |                                                                                                                                                                                                                 |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Reads ahead <i>nPages</i> starting at <i>startPage</i> (if the file is linearized).                                                                                                                             |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document for which pages are read ahead.</p> <p><i>startPage</i></p> <p>The page for which read ahead is initiated.</p> <p><i>nPages</i></p> <p>The number of pages to read ahead.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                            |
| <b>Exceptions</b>      | None                                                                                                                                                                                                            |
| <b>Notifications</b>   | None                                                                                                                                                                                                            |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                |
| <b>Related Methods</b> | None                                                                                                                                                                                                            |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                        |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDDocRelease

```
void PDDocRelease (PDDoc doc);
```

|                        |                                                                                                                                            |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Decrements a document's reference count. The document will not be closed until the reference count is zero, or the application terminates. |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document whose reference count is decremented.</p>                                                                |
| <b>Return Value</b>    | None                                                                                                                                       |
| <b>Exceptions</b>      | <a href="#">genErrBadUnlock</a>                                                                                                            |
| <b>Notifications</b>   | None                                                                                                                                       |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                           |
| <b>Related Methods</b> | <a href="#">PDDocAcquire</a>                                                                                                               |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                   |

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDDocRemoveNameTree

```
void PDDocRemoveNameTree (PDDoc thePDDoc, ASAtom theTree);
```

|                        |                                                                                                                                 |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Removes the name tree inside the Names dictionary with the specified key name. Does nothing if no object with that name exists. |
| <b>Parameters</b>      | <p><i>thePDDoc</i><br/>The document from which a name tree is removed.</p> <p><i>theTree</i><br/>The name tree to remove.</p>   |
| <b>Return Value</b>    | None                                                                                                                            |
| <b>Exceptions</b>      | None                                                                                                                            |
| <b>Notifications</b>   | None                                                                                                                            |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                |
| <b>Related Methods</b> | <a href="#">ASAtomFromString</a><br><a href="#">PDDocCreateNameTree</a><br><a href="#">PDDocGetNameTree</a>                     |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                        |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## PDDocRemovePageLabel

```
void PDDocRemovePageLabel (PDDoc pdDoc, ASInt32 pageNum);
```

|                        |                                                                                                                                                         |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Removes the page label that is attached to the specified page, effectively merging the specified range with the previous page label sequence.           |
| <b>Parameters</b>      | <p><i>pdDoc</i></p> <p>The document from which a page label is removed.</p> <p><i>pageNum</i></p> <p>The page from which the page label is removed.</p> |
| <b>Return Value</b>    | None                                                                                                                                                    |
| <b>Exceptions</b>      | None                                                                                                                                                    |
| <b>Notifications</b>   | None                                                                                                                                                    |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                        |
| <b>Related Methods</b> | <a href="#">PDDocSetPageLabel</a><br><a href="#">PDDocGetPageLabel</a><br><a href="#">PDPageLabelNew</a>                                                |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | —         | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | —         | M, W    |
| Reader            | —         | —       |

## PDDocRemoveStructTreeRoot

```
void PDDocRemoveStructTreeRoot (PDDoc pdDoc);
```

|                        |                                                                                                                     |
|------------------------|---------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Removes, but does <i>not</i> destroy, the specified <i>StructTreeRoot</i> element from the specified <i>PDDoc</i> . |
| <b>Parameters</b>      | <p><i>pdDoc</i></p> <p>The <i>PDDoc</i> for which the <i>StructTreeRoot</i> element is removed.</p>                 |
| <b>Return Value</b>    | None                                                                                                                |
| <b>Exceptions</b>      | None                                                                                                                |
| <b>Notifications</b>   | None                                                                                                                |
| <b>Header File</b>     | <i>PDSWriteCalls.h</i>                                                                                              |
| <b>Related Methods</b> | <a href="#">PDDocCreateStructTreeRoot</a><br><a href="#">PDDocGetStructTreeRoot</a>                                 |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.          |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDDocRemoveThread

```
void PDDocRemoveThread (PDDoc doc, ASInt32 index);
```

**Description** Removes an article thread from a document. If you also wish to destroy the thread, use [PDThreadDestroy](#) after calling *PDDocRemoveThread*.

**Parameters**

*doc*  
The document from which the thread is removed.

*index*  
The index of the thread to remove.

**Return Value** None

**Exceptions** None

**Notifications** [PDDocDidRemoveThread](#)  
[PDDocDidRemoveThread](#)

**Header File** *PDCalls.h*

**Related Methods** [PDThreadDestroy](#)  
[PDDocAddThread](#)  
[PDBeadGetThread](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0  |
|-------------|-----------|------|
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDDocReplacePages

```
void PDDocReplacePages (PDDoc doc, ASInt32 startPage,
 PDDoc doc2, ASInt32 startPageDoc2, ASInt32 numPages,
 ASBool mergeTextAnnots, ProgressMonitor progMon,
 void* progMonClientData, CancelProc cancelProc,
 void* cancelProcClientData);
```

### Description

Replaces the specified range of pages in one document with pages from another. The contents, resources, size and rotation of the pages are replaced. Bookmarks are *not* copied, because they are attached to the document, not to individual pages.

*Note: Annotations in the replaced pages are not replaced and remain with the page. Use [PDDocDeletePages](#) to remove annotations.*

### Parameters

*doc*

The document in which pages are replaced.

*startPage*

The first page number in *doc* to replace. The first page is 0.

*doc2*

The document from which pages are copied into *doc*.

*startPageDoc2*

The page number of the first page in *doc2* to copy. The first page is 0.

*numPages*

The number of pages to replace.

*mergeTextAnnots*

If *true*, text annotations from *doc2* are appended if they are different than all existing annotations on the page in *doc*. No other types of annotations are copied.

*progMon*

A progress monitor. Use [AVAppGetDocProgressMonitor](#) to obtain the default progress monitor. *NULL* may be passed, in which case no progress monitor is used.

## *progMonClientData*

Pointer to user-supplied data to pass to *progMon* each time it is called. Should be *NULL* if *progMon* is *NULL*.

## *cancelProc*

Currently unused.

A cancel procedure. Use [AVAppGetCancelProc](#) to obtain the current cancel procedure. May be *NULL*, in which case no cancel proc is used.

## *cancelProcClientData*

Pointer to user-supplied data to pass to *cancelProc* each time it is called. Should be *NULL* if *cancelProc* is *NULL*.

## Return Value

None

## Exceptions

Among others:

Raises [pdErrOpNotPermitted](#) unless *doc* is editable and *doc2* is not encrypted or the owner opened it.

Raises [pdErrCantUseNewVersion](#) if *doc2* is a newer major version than the Acrobat viewer understands.

Raises [genErrBadParm](#) if *numPages* < 1, 1 *startPage* < 0, *startPage* + *numPages* is greater than the number of pages in *doc*, *startPageDoc2* < 0, or *startPageDoc2* + *numPages* is greater than the number of pages in *doc2*.

Raises [genErrNoMemory](#) if there is insufficient memory to perform the insertion.

## Notifications

[PDDocWillReplacePages](#)  
[PDDocDidReplacePages](#)  
[PDDocDidChangePages](#)  
[PDDocWillChangePages](#)

## Header File

*PDCalls.h*

## Related Methods

[PDDocImportNotes](#)  
[PDDocMovePage](#)  
[PDDocDeletePages](#)  
[PDDocCreatePage](#)

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDDocSave

```
void PDDocSave (PDDoc doc, PDSaveFlags saveFlags,
 ASPathName newPath, ASFileSys fileSys, ProgressMonitor progMon,
 void* progMonClientData);
```

### Description

Saves a document to disk. If a full save is requested to the original path, the file is saved to a file system-determined temporary file, the old file is deleted, and the temporary file is renamed to *newPath*. You must call [PDDocClose](#) to release resources; do *not* call [PDDocRelease](#).

If the document was created with [PDDocCreate](#), at least one page must be added using [PDDocCreatePage](#) or [PDDocImportNotes](#) before the Acrobat viewer can save the document.

You can replace this method with your own version, using [HFTReplaceEntry](#).

### Parameters

*doc*

The document to save.

*saveFlags*

A bit field composed of an OR of the [PDSaveFlags](#) values.

*newPath*

The path to which the file is saved. A path must be specified when either *PDSaveFull* or *PDSaveCopy* are used for *saveFlags*. If *PDSaveIncremental* is specified in *saveFlags*, then *newPath* should be *NULL*.

If *PDSaveFull* is specified and *newPath* is the same as the file's original path, the new file is saved to a file system-determined temporary path, then the old file is deleted and the new file is renamed to *newPath*.



## *fileSys*

File system. If *NULL*, uses the *fileSys* of the document's current backing file.

*Note: Files can only be saved to the same file system. fileSys must be either NULL or the default file system obtained with [ASGetDefaultFileSys](#), otherwise an error is raised.*

## *progMon*

Progress monitor. Use [AVAppGetDocProgressMonitor](#) to obtain the default. *NULL* may be passed, in which case no progress monitor is used.

## *progMonClientData*

Pointer to user-supplied data to pass to *progMon* each time it is called. Should be *NULL* if *progMon* is *NULL*.

## Return Value

None

## Exceptions

Among others:

Raises [pdErrAfterSave](#) if the save was completed successfully, but there were problems cleaning up afterwards. The document is no longer consistent and cannot be changed. It must be closed and reopened.

Raises [pdErrOpNotPermitted](#) if saving is not permitted. Saving is permitted if either edit or *editNotes* (see [PDPerms](#)) is allowed, or you are doing a full save and *saveAs* is allowed.

Raises [pdErrAlreadyOpen](#) if *PDSaveFull* is used, and the file specified by *newPath* is already open.

## Notifications

[PDDocWillSave](#)  
[PDDocDidSave](#)

## Header File

*PDCalls.h*

## Related Methods

[PDDocClose](#)  
[PDDocSaveWithParams](#)

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDDocSaveWithParams

```
void PDDocSaveWithParams (PDDoc doc,
 PDDocSaveParams inParams);
```

### Description

Saves a document to disk as specified in a parameter's structure. This is essentially the same as [PDDocSave](#) with two additional parameters: a cancel proc and cancel proc client data (so you could cut and paste description information, and so on, from [PDDocSave](#)). You can replace this method with your own version, using [HFTReplaceEntry](#).

### Parameters

*doc*

The document to save.

*inParams*

[PDDocSaveParams](#) structure specifying how the document should be saved.

### Return Value

None

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDDocSave](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | —         | —       |

## PDDocSetFlags

```
void PDDocSetFlags (PDDoc doc, ASInt32 flags);
```

**Description** Sets information about the document's file and its state. This method can only be used to set, not clear, flags. As a result, it is not possible, for example, to use this method to clear the flag that indicates that a document has been modified and needs to be saved. Instead, use [PDDocClearFlags](#) to clear flags.

**Parameters**

*doc*  
The document whose flags are set.

*flags*  
A bit field composed of an OR of the [PDDocFlags](#) values.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocGetFlags](#)  
[PDDocClearFlags](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDDocSetFullScreen

```
void PDDocSetFullScreen (PDDoc pdDoc, ASBool fs);
```

**Description** Sets whether or not this document opens in full-screen mode. This provides an alternative to calling [PDDocSetPageLabel](#) with *PDFFullScreen*.

**Parameters**

*pdDoc*  
The document to set.

*fs*  
*true* if the document is set to open in full-screen mode, *false* otherwise.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocGetFullScreen](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | —         | —       |

## PDDocSetInfo

```
void PDDocSetInfo (PDDoc doc, const char* infoKey,
 char* buffer, ASInt32 nBytes);
```

### Description

Sets the value of a key in a document's Info dictionary. See Section 6.10 on info dictionaries in the [Portable Document Format Reference Manual](#) for information about Info dictionaries. All values in the Info dictionary should be strings; other data types such as numbers and booleans should not be used as values in the Info dictionary. If an info dictionary key is specified that is not currently in the info dictionary, it is added to the dictionary.

Users may define their own Info dictionary entries. In this case, it is strongly recommended that the key have the developer's prefix assigned by the Adobe Developers Association.

*Note: For Roman viewers, this text is always stored in the PDFDocEncoding. For non-Roman character set viewers, this text is stored as PDFDocEncoding or Unicode, depending on the file's creator. Files created in a non-Roman environment contain Unicode versions of these strings; in a Roman environment, files contain PDFDocEncoding versions of these strings.*

### Parameters

*doc*

The document whose Info dictionary key is set.

*infoKey*

The name of the Info dictionary key whose value is set.

*buffer*

Buffer containing the value to associate with *infoKey*.

*nBytes*

The number of bytes in *buffer*.

### Return Value

None

### Exceptions

None

### Notifications

None

**Header File** *PDCalls.h*

**Related Methods** [PDDocGetInfo](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDDocSetNewCryptHandler

```
void PDDocSetNewCryptHandler (PDDoc pdDoc,
 ASAtom newCryptHandler);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | <p>Sets specified document's new security handler (that is, the security handler that will be used after the document is saved).</p> <p>This method simply returns if the new security handler is the same as the old one. Otherwise, the new security handler's <a href="#">PDCryptNewSecurityDataProc</a> is called to set the document's <i>newSecurityData</i> field. If the new security handler does not have <a href="#">PDCryptNewSecurityDataProc</a>, the document's <i>newSecurityData</i> field is set to 0.</p> |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document whose new security handler is set.</p> <p><i>newCryptHandler</i></p> <p>The <i>ASAtom</i> corresponding to the name of the new security handler to use for the document. This name must be the same as the <i>pdfName</i> used when the security handler was registered using <a href="#">PDRegisterCryptHandler</a>. Use <i>ASAtomNull</i> to remove security from the document.</p>                                                                                                      |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>Exceptions</b>      | <p>Raises <a href="#">pdErrNoCryptHandler</a> if there is no security handler registered with the specified name and the name is not <i>ASAtomNull</i>.</p> <p>Raises <a href="#">pdErrOpNotPermitted</a> if the document's permissions do not allow its security to be modified.</p>                                                                                                                                                                                                                                        |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>Header File</b>     | <i>PICrypt.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>Related Methods</b> | <a href="#">PDDocGetNewCryptHandler</a><br><a href="#">PDRegisterCryptHandler</a>                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Available in:</b>   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |



|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDDocSetNewSecurityData

```
void PDDocSetNewSecurityData (PDDoc doc, void* secData);
```

|                        |                                                                                                                                                                                                                                                                                 |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Sets the security data structure for the specified document's new security handler. Use <a href="#">PDDocSetNewCryptHandler</a> to set a new security handler for a document.                                                                                                   |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document whose new security data structure is set.</p> <p><i>secData</i></p> <p>Pointer to the new security data structure to set for <i>doc</i>. See <a href="#">PDDocNewSecurityData</a> for information on creating and filling this structure.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                            |
| <b>Exceptions</b>      | <p>Raises <a href="#">pdErrNeedCryptHandler</a> if the document does not have a new security handler.</p> <p>Raises <a href="#">pdErrOpNotPermitted</a> if the document's permissions cannot be changed.</p>                                                                    |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                            |
| <b>Header File</b>     | <i>PICrypt.h</i>                                                                                                                                                                                                                                                                |
| <b>Related Methods</b> | <a href="#">PDDocGetNewSecurityData</a><br><a href="#">PDDocGetSecurityData</a><br><a href="#">PDDocNewSecurityData</a><br><a href="#">PDDocSetNewCryptHandler</a>                                                                                                              |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                        |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | —         | —       |



## PDDocSetOpenAction

```
void PDDocSetOpenAction (PDDoc doc, PDAction action);
```

**Description** Sets the value of the *OpenAction* key in the Catalog dictionary, which is the action performed when the document is opened.

**Parameters** *doc*  
The document whose open action is set.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocGetOpenAction](#)  
[PDActionFromCosObj](#)  
[PDActionNew](#)  
[PDActionNewFromDest](#)  
[PDActionNewFromFileSpec](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | —         | —       |

## PDDocSetPageLabel

```
void PDDocSetPageLabel (PDDoc pdDoc, ASInt32 pageNum,
 PDPageLabel pgLabel);
```

**Description** Attaches a label to a page. This establishes the numbering scheme for that page and all following it, until the next page with an attached label is encountered. This label allows PDF producers to define a page numbering system other than the Acrobat default.

**Parameters**

*pdDoc*  
The document containing the page to label.

*pageNum*  
The number of the page to label.

*pgLabel*  
Label for the page specified by pageNum.

**Return Value** If pageNum is less than 0 or greater than the number of pages in pdDoc, nothing is done.

**Exceptions** Raises [genErrBadParm](#) if *pgLabel* is not a valid *PDPageLabel*.

**Notifications** [PDDocPageLabelDidChange](#)

**Header File** *PDCalls.h*

**Related Methods** [PDDocFindPageNumForLabel](#)  
[PDDocGetPageLabel](#)  
[PDDocGetLabelForPageNum](#)  
[PDDocRemovePageLabel](#)

**Example**

```
PDPageLabel pgLabel;
/* Start page numbering with page 5 */
PDDocSetPageLabel (pdDoc, 5L, pgLabel);
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

**Available in:**

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0  |
|-------------|-----------|------|
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## PDDocSetPageMode

```
void PDDocSetPageMode (PDDoc doc, PDPageMode mode);
```

|                        |                                                                                                               |
|------------------------|---------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Sets the value of the <i>PageMode</i> key in the Catalog dictionary.                                          |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document whose page mode is set.</p> <p><i>mode</i></p> <p>The page mode to set.</p> |
| <b>Return Value</b>    | None                                                                                                          |
| <b>Exceptions</b>      | None                                                                                                          |
| <b>Notifications</b>   | None                                                                                                          |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                              |
| <b>Related Methods</b> | <a href="#">PDDocGetPageLabel</a><br><a href="#">AVDocSetViewMode</a>                                         |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.      |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | —         | —       |

**PDFFileSpec**



## PDFFileSpecAcquireASPath

```
ASPathName PDFFileSpecAcquireASPath (PDFFileSpec fileSpec,
 ASPathName relativeToThisPath);
```

- Description** Acquires an *ASPathName* for the specified file specification and relative path.
- Parameters**
- fileSpec*  
The file specification for which an *ASPathName* is acquired.
- relativeToThisPath*  
A pathname relative to which the *fileSpec* is interpreted. If *NULL*, *fileSpec* is assumed to be an absolute, not a relative, path.
- Return Value** The *ASPathName* corresponding to *fileSpec*.  
After you are done using the *ASPathName*, you must free it using [ASFileSysReleasePath](#).
- Exceptions** None
- Notifications** None
- Header File** *PDCalls.h*
- Related Methods** [ASFileSysReleasePath](#)
- Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFFileSpecFromCosObj

```
PDFFileSpec PDFFileSpecFromCosObj (CosObj obj);
```

**Description** Converts an appropriate string or dictionary Cos object to a file specification. This method does not copy the object, but is instead the logical equivalent of a type cast.

**Parameters** *obj*  
The Cos object to convert to a file specification.

**Return Value** The file specification corresponding to *obj*.

**Exceptions** Raises [pdErrBadFileSpec](#) if the file specification is not valid, as determined by [PDFFileSpecIsValid](#).

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDFFileSpecGetCosObj](#)  
[PDFFileSpecIsValid](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFFileSpecGetCosObj

[CosObj](#) PDFFileSpecGetCosObj ([PDFFileSpec](#) fileSpec);

|                        |                                                                                                                                                       |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the Cos object associated with a file specification. This method does not copy the object, but is instead the logical equivalent of a type cast. |
| <b>Parameters</b>      | <i>fileSpec</i><br>The file specification whose Cos object is obtained.                                                                               |
| <b>Return Value</b>    | The string or dictionary Cos object corresponding to the file specification.                                                                          |
| <b>Exceptions</b>      | None                                                                                                                                                  |
| <b>Notifications</b>   | None                                                                                                                                                  |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                      |
| <b>Related Methods</b> | <a href="#">PDFFileSpecFromCosObj</a>                                                                                                                 |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                              |

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFFileSpecGetDIPath

```
ASInt32 PDFFileSpecGetDIPath (PDFFileSpec fileSpec, char* buffer,
 ASInt32 bufLen);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the device-independent pathname from a file specification.                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>Parameters</b>      | <p><i>fileSpec</i></p> <p>The file specification whose device-independent pathname is obtained.</p> <p><i>buffer</i></p> <p>(Filled by the method) <i>NULL</i>-terminated device-independent pathname. If <i>buffer</i> is <i>NULL</i>, the method simply returns the length of the pathname.</p> <p><i>bufLen</i></p> <p>Length of <i>buffer</i>, in bytes. If the device-independent pathname is longer than this, only the first <i>bufLen</i> – 1 bytes are copied into <i>buffer</i>, plus a <i>NULL</i> at the end of the buffer.</p> |
| <b>Return Value</b>    | Number of characters (excluding the <i>NULL</i> ) copied into <i>buffer</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>Related Methods</b> | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                                                                                                                                                                                                    |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDFFileSpecGetDoc

```
PDDoc PDFFileSpecGetDoc (PDFFileSpec fileSpec);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the <i>PDDoc</i> that contains <i>fileSpec</i> .                                                    |
| <b>Parameters</b>      | <i>fileSpec</i><br>A <i>PDFFileSpec</i> in a document.                                                   |
| <b>Return Value</b>    | A <i>PDDoc</i> or <i>NULL</i> if the file specification <i>CosObj</i> is not in a document.              |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | None                                                                                                     |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher. |

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFFileSpecGetFileSys

```
ASFileSys PDFFileSpecGetFileSys (PDFFileSpec fileSpec);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the file system that services the specified file specification.                                     |
| <b>Parameters</b>      | <p><i>fileSpec</i></p> <p>The file specification whose file system is obtained.</p>                      |
| <b>Return Value</b>    | The file system that services <i>fileSpec</i> .                                                          |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDFFileSpecGetFileSysName</a>                                                                |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher. |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFFileSpecGetFileSysName

[ASAtom](#) PDFFileSpecGetFileSysName ( [PDFFileSpec](#) fileSpec );

### Description

Gets the name of the file system that a *PDFFileSpec* belongs to. For a simple *fileSpec* (string form), the name of the file system is the name of the document's file system—if the *CosObj* that is the *fileSpec* is contained in a document. For a complex *fileSpec* (dict form) with an */FS* key, the name of the file system is the atom associated with the **FS** key.

The file system returned by [PDFFileSpecGetFileSys](#) is the file system that has registered a [PDFFileSpecHandler](#) for *fileSpec*'s file system NAME (if there is one), and is not necessarily the same as [ASFileGetFileSysByName](#)([PDFFileSpecGetFileSysName](#)(*fileSpec*));

### Parameters

*fileSpec*  
A *PDFFileSpec*.

### Return Value

An *ASAtom* representing the file system of *fileSpec*.

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDFFileSpecGetFileSys](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDFFileSpecIsValid

```
ASBool PDFFileSpecIsValid (PDFFileSpec fileSpec);
```

**Description** Tests whether or not a file specification is valid. This is intended only to ensure that the file spec has not been deleted, not to ensure that all necessary information is present and valid.

**Parameters** *fileSpec*  
The file specification whose validity is tested.

**Return Value** *true* if *fileSpec* is valid, *false* otherwise.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** None

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFFileSpecNewFromASPath

```
PDFFileSpec PDFFileSpecNewFromASPath (PDDoc pdDoc,
 ASFileSys fileSys, ASPathName path,
 ASPathName relativeToThisPath);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Creates a new file specification from the specified <i>ASPathName</i> , using the <a href="#">PDFFileSpecNewFromASPathProc</a> of the specified file system's file specification handler.                                                                                                                                                                                                                                                                                                         |
| <b>Parameters</b>      | <i>pdDoc</i><br>The document in which the new file specification will be used.<br><br><i>fileSys</i><br>Pointer to an <a href="#">ASFileSysRec</a> specifying the file system responsible for the newly-created file specification.<br><br><i>path</i><br>The path to convert into a file specification.<br><br><i>relativeToThisPath</i><br>Pathname relative to which <i>path</i> is interpreted. If <i>NULL</i> , <i>path</i> is interpreted as an absolute pathname, not a relative pathname. |
| <b>Return Value</b>    | The newly-created file spec, or an invalid file spec if the <i>ASPathName</i> cannot be converted to a <i>PDFFileSpec</i> (use <a href="#">PDFFileSpecsValid</a> to test whether or not the conversion was successful).                                                                                                                                                                                                                                                                           |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Related Methods</b> | <a href="#">PDFFileSpecsValid</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                                                                                                                                                          |

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0  |
|-------------|-----------|------|
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

**PDFont**

## PDFontAcquireEncodingArray

```
ASUns8** PDFontAcquireEncodingArray (PDFont font);
```

### Description

Acquires a font's encoding array (the mapping of character codes to glyphs). When done with this array, call [PDFontEncodingArrayRelease](#) to release it.

The array contains 256 pointers. If a pointer is not *NULL*, it points to a C string containing the name of the glyph for the code point corresponding to the index. If it is *NULL*, then the name of the glyph is unchanged from that specified by the font's built-in encoding.

For a Type 3 font, all glyph names will be present in the encoding array, and *NULL* entries correspond to unencoded code points.

For non-Roman character set viewers, it is *not* appropriate to call this method.

### Parameters

*font*

The font whose encoding array is obtained.

### Return Value

The font's encoding array. Returns *NULL* if there is no encoding array associated with the font.

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDFontEncodingArrayRelease](#)  
[PDFontGetEncodingIndex](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDFontAcquireXlateTable

```
ASInt16* PDFontAcquireXlateTable (PDFont font);
```

**Description**

Increments the specified font's *XlateTable* reference count and also returns the *XlateTable*, which is a 256-entry table that maps characters from their encoding in the PDF file to host encoding. If a character cannot be mapped to host encoding, then the table entry will (for that character) contain -1. When you are done using the *XlateTable*, call [PDFontXlateTableRelease](#) to release it.

For non-Roman character set viewers, it is *not* appropriate to call this method.

**Parameters**

*font*

The font whose *XlateTable* is obtained.

**Return Value**

Pointer to the font's *XlateTable*, if any. Otherwise *NULL*.

**Exceptions**

None

**Notifications**

None

**Header File**

*PDCalls.h*

**Related Methods**

[PDFontXlateTableRelease](#)  
[PDFontXlateString](#)

**Availability**

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontEncodingArrayRelease

```
void PDFontEncodingArrayRelease (ASUns8** array);
```

|                        |                                                                                                                                                                                                      |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Releases a font's encoding array (the mapping of character codes to glyphs). Call this method after you are done using an encoding array acquired using <a href="#">PDFontAcquireEncodingArray</a> . |
| <b>Parameters</b>      | <i>array</i><br>The encoding array to release.                                                                                                                                                       |
| <b>Return Value</b>    | None                                                                                                                                                                                                 |
| <b>Exceptions</b>      | None                                                                                                                                                                                                 |
| <b>Notifications</b>   | None                                                                                                                                                                                                 |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                     |
| <b>Related Methods</b> | <a href="#">PDFontAcquireEncodingArray</a>                                                                                                                                                           |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                             |

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDFontEnumCharProcs

```
void PDFontEnumCharProcs (PDFont font, PDCharProcEnumProc proc,
 void* clientData);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Enumerates a Type 3 font's character drawing procedures. The elements of a single character procedure can be enumerated using <a href="#">PDCharProcEnum</a> .                                                                                                                                                                                                                                                                       |
| <b>Parameters</b>      | <p><i>font</i></p> <p>The Type 3 font's character drawing procedures are being enumerated.</p> <p><i>proc</i></p> <p>User-supplied callback to call for each character in the font.</p> <p>Enumeration ends if <i>proc</i> returns <i>false</i>. If the font contains no characters, <i>proc</i> will not be called.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data passed to <i>proc</i> each time it is called.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Related Methods</b> | <a href="#">PDCharProcEnum</a><br><a href="#">PDDocEnumFonts</a><br><a href="#">PDDocEnumLoadedFonts</a>                                                                                                                                                                                                                                                                                                                             |
| <b>Example</b>         | <pre>ACCB1 ASBool ACCB2 myPDCharProcEnumProc(     char* name, PDCharProc obj,     void* clientData) {     ... } PDFontEnumCharProcs(font,     ASCallbackCreateProto(         PDCharProcEnumProc,         myPDCharProcEnumProc), NULL);</pre>                                                                                                                                                                                         |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                                                                                             |

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontFromCosObj

```
PDFont PDFontFromCosObj (CosObj fontObj);
```

|                        |                                                                                                                                         |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Converts a dictionary Cos object to a font. This method does not copy the object, but is instead the logical equivalent of a type cast. |
| <b>Parameters</b>      | <i>fontObj</i><br>Dictionary Cos object for the font.                                                                                   |
| <b>Return Value</b>    | The <i>PDFont</i> for <i>fontObj</i> . Returns <i>NULL</i> if there is no <i>CosDoc</i> or <i>PDDoc</i> containing <i>fontObj</i> .     |
| <b>Exceptions</b>      | None                                                                                                                                    |
| <b>Notifications</b>   | None                                                                                                                                    |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                        |
| <b>Related Methods</b> | None                                                                                                                                    |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDFontGetBBox

```
void PDFontGetBBox (PDFont font, ASFixedRect* bboxP);
```

**Description** Gets a Type 3 font's bounding box, which is the smallest rectangle that would enclose every character in the font if they were overlaid and painted.

**Parameters**

*font*

The font whose bounding box is obtained.

*bboxP*

(Filled by the method) Pointer to a rectangle specifying the font's bounding box.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** None

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFFontGetCharSet

[PDCharSet](#) PDFFontGetCharSet ([PDFont](#) font);

**Description** Gets the font's character set. This is derived from the "Uses Adobe standard encoding" bit in the font descriptor (if the font has a font descriptor) or from the font's name (if the font is one of the base 14 fonts and does not have a font descriptor).

For non-Roman character set viewers, call [PDFFontGetEncodingName](#) instead.

**Parameters** *font*  
The font whose character set is obtained.

**Return Value** The font's character set. For non-Roman character set viewers, returns *PDUnknownCharSet*.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDFFontGetEncodingName](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontGetCIDSystemInfo

[ASAtom](#) PDFontGetCIDSystemInfo ([PDFont](#) font);

### Description

Gets an *ASAtom* representing Registry and Ordering for a CIDFont. This information resides in the **CIDSystemInfo** entry of the CIDFont dictionary, which describes a CIDFont.

*PDFontGetCIDSystemInfo* takes either a Type 0 font or descendant font (CIDType0 or CIDType2) as an argument. This information is always present for any Type 0 font; the actual registry ordering information is a part of the descendant font.

This method provides one way to identify a font's language.

The **CIDSystemInfo** entry uses three components to identify a character collection uniquely:

- A registry name to identify an issuer of orderings.
- An ordering name to identify an ordered character collection.
- A supplement number to indicate that the ordered character collection for a registry, ordering, and previous supplement has been changed to add new characters assigned CIDs beginning with the next available CID.

The *PDFontGetCIDSystemInfo* method obtains the first two of these components.

A CIDFont is designed to contain a large number of glyph procedures. Instead of being accessed by a name, each glyph procedure is accessed by an integer known as a character identifier or CID. Instead of a font encoding, CIDFonts use a CMap with a Type 0 composite font to define the mapping from character codes to a font number and a character selector.

For more information on Type 0 fonts, CIDFonts, and CMaps, see Section 7.7.9 in the [Portable Document Format Reference Manual](#). For detailed information on CIDFonts, see Technical Note #5092, *CID-Keyed Font Technology Overview*, and Technical Note #5014, *Adobe CMap and CIDFont Files Specification*.

|                        |                                                                                                              |
|------------------------|--------------------------------------------------------------------------------------------------------------|
| <b>Parameters</b>      | <i>font</i><br><br>The font whose Registry and Ordering information is obtained.                             |
| <b>Return Value</b>    | The <i>ASAtom</i> representing the CIDFont's Registry and Ordering information, for example, "Adobe-Japan1." |
| <b>Exceptions</b>      | None                                                                                                         |
| <b>Notifications</b>   | None                                                                                                         |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                             |
| <b>Related Methods</b> | <a href="#">PDFontGetCIDSystemSupplement</a><br><a href="#">PDFontGetDescendant</a>                          |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020003 or higher.     |

## Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDFontGetCIDSystemSupplement

```
ASInt32 PDFontGetCIDSystemSupplement (PDFont font);
```

### Description

Gets the SystemSupplement number of a CIDFont. This field resides in the **CIDSystemInfo** entry of the CIDFont dictionary, which describes a CIDFont.

The **CIDSystemInfo** entry uses three components to identify a character collection uniquely:

- A registry name to identify an issuer of orderings.
- An ordering name to identify an ordered character collection.
- A supplement number to indicate that the ordered character collection for a registry, ordering, and previous supplement has been changed to add new characters assigned CIDs beginning with the next available CID.

[PDFontGetCIDSystemInfo](#) provides character collection information, and [PDFontGetCIDSystemSupplement](#) specifies the version of the ordering.

A CIDFont is designed to contain a large number of glyph procedures. Instead of being accessed by a name, each glyph procedure is accessed by an integer known as a character identifier or CID. Instead of a font encoding, CIDFonts use a CMap with a Type 0 composite font to define the mapping from character codes to a font number and a character selector.

For more information on Type 0 fonts, CIDFonts, and CMaps, see Section 7.7.9 in the [Portable Document Format Reference Manual](#). For detailed information on CIDFonts, see Technical Note #5092, *CID-Keyed Font Technology Overview*, and Technical Note #5014, *Adobe CMap and CIDFont Files Specification*.

### Parameters

*font*

The font whose SystemSupplement field is obtained.

### Return Value

The SystemSupplement field from the CIDFont.

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*



**Related Methods**    [PDFontGetDescendant](#)  
[PDFontGetCIDSystemInfo](#)

**Availability**        Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontGetCosObj

[CosObj](#) PDFontGetCosObj ([PDFont](#) font);

|                        |                                                                                                                             |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the Cos object for a font. This method does not copy the object, but is instead the logical equivalent of a type cast. |
| <b>Parameters</b>      | <p><i>font</i></p> <p>The font whose Cos object is obtained.</p>                                                            |
| <b>Return Value</b>    | The dictionary Cos object for the font. Its contents may be enumerated using <a href="#">CosObjEnum</a> .                   |
| <b>Exceptions</b>      | None                                                                                                                        |
| <b>Notifications</b>   | None                                                                                                                        |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                            |
| <b>Related Methods</b> | <a href="#">PDDocEnumFonts</a>                                                                                              |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                    |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontGetDescendant

[PDFont](#) PDFontGetDescendant ([PDFont](#) font);

### Description

Gets a Type 0 font's descendant, which may be a CIDType0 or CIDType2 font.

Type 0 fonts support single-byte or multibyte encodings and can refer to one or more *descendant fonts*. These fonts are analogous to the Type 0 or composite fonts supported by Level 2 PostScript interpreters. However, PDF Type 0 fonts only support character encodings defined by a CMap. The CMap specifies the mappings between character codes and the glyphs in the descendant fonts.

For information on Type 0 fonts, see Section 7.7.7 in the [Portable Document Format Reference Manual](#). See Section 7.9 for more details on CMaps.

### Parameters

*font*

The font whose descendant is obtained.

### Return Value

The font's descendant font.

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDFontGetCIDSystemInfo](#)  
[PDFontGetCIDSystemSupplement](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontGetEncodingIndex

```
ASInt32 PDFontGetEncodingIndex (PDFont font);
```

|                        |                                                                                                                                                                                                                                                                  |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets a font's encoding index.<br>For non-Roman character set viewers, it is <i>not</i> appropriate to call this method. Call <a href="#">PDFontGetEncodingName</a> instead.                                                                                      |
| <b>Parameters</b>      | <i>font</i><br><br>The font whose encoding index is obtained.                                                                                                                                                                                                    |
| <b>Return Value</b>    | A font encoding index. If the index is greater than <i>PDLastKnownEncoding</i> , it is a custom encoding, and is unique within the document. If the index is less than <i>PDLastKnownEncoding</i> , it must be one of the <a href="#">PDFontEncoding</a> values. |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                             |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                             |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                 |
| <b>Related Methods</b> | <a href="#">PDFontAcquireEncodingArray</a>                                                                                                                                                                                                                       |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                         |

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFFontGetEncodingName

```
const ASUns8* PDFFontGetEncodingName (PDFFont font);
```

### Description

Gets a string representing a font's encoding.

Use [PDFFontGetEncodingIndex](#) to get encoding information for Roman viewers. In Roman systems, a font encoding describes a font's character encoding—the mapping between numeric character codes and character names. It may be MacRomanEncoding in Mac OS, WinAnsiEncoding in Windows, ISO8859-1 (ISO Latin-1) in UNIX, or some other encoding, such as SHIFT-JIS on Japanese systems. See Section 7.8 in the [Portable Document Format Reference Manual](#) for information on font encodings. See Appendix C in the [Portable Document Format Reference Manual](#) for descriptions of MacRomanEncoding and WinAnsiEncoding.

For non-Roman systems, the font encoding may be a variety of encodings, which are defined by a CMap (character map). See Section 7.9 in the [Portable Document Format Reference Manual](#) for a list of predefined CMaps. In this case, the encoding string contains values such as "90ms-RKSJ-H", "90msp-RKSJ-H", "Identity-V", or "90pv-RKSJ-H"; it doesn't return a string like "Shift-JIS".

### Parameters

*font*

The font whose encoding name is obtained.

### Return Value

String representing the font's encoding.

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDFFontGetEncodingIndex](#)  
[PDGetHostEncoding](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontGetFontMatrix

```
void PDFontGetFontMatrix (PDFont fontP, ASFixedMatrix* matrixP);
```

**Description** Gets a font's matrix, which specifies the transformation from character space to text space. See Section 3.10 in the [Portable Document Format Reference Manual](#). This is only valid for Type 3 fonts.

**Parameters**

*font*  
The font whose matrix is obtained.

*matrixP*  
(Filled by the method) Pointer to the font's matrix.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocEnumFonts](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontGetMetrics

```
void PDFontGetMetrics (PDFont font, PDFontMetricsP metricsP,
 os_size_t sizeMetrics);
```

**Description** Gets a font's metrics, which provide the information needed to create a substitute multiple master font when the original font is unavailable. See Section 7.10 in the [Portable Document Format Reference Manual](#) for a discussion of font descriptors.

**Parameters**

*font*  
The font whose metrics are being obtained.

*metricsP*  
(Filled by the method) Pointer to a [PDFontMetrics](#) structure containing the font's metrics.

*sizeMetrics*  
Must be *sizeof(PDFontMetrics)*.

**Return Value** None

**Notifications** None

**Exceptions** None

**Header File** *PDCalls.h*

**Related Methods** [PDFontGetWidths](#)  
[PDFontSetMetrics](#)

**Example**

```
PDWordFinder wObj;
PDWord pdWord;
PDStyle style;
PDFont font;
PDFontMetrics fMetrics;

/* Get font metric info */
style = PDWordGetNthCharStyle(wObj, pdWord,
 0);
font = PDStyleGetFont(style);
PDFontGetMetrics (font, &fMetrics,
 sizeof(PDFontMetrics));
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**



|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontGetName

```
ASInt32 PDFontGetName (PDFont font, char* buffer,
 ASInt32 bufSize);
```

|                        |                                                                                                                                                                                                                                                                                                                                          |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the name of a font. The behavior depends on the font type; for a Type 3 font it gets the value of the <i>Name</i> key in a PDF Font resource. See Section 7.7 in the <a href="#">Portable Document Format Reference Manual</a> . For other types it gets the value of the <i>BaseFont</i> key in a PDF font resource.               |
| <b>Parameters</b>      | <p><i>font</i></p> <p>The font whose name is obtained.</p> <p><i>buffer</i></p> <p>(Filled by the method) Buffer into which the font's name is stored.</p> <p><i>bufSize</i></p> <p>Length of <i>buffer</i>, in bytes. The maximum font name length that the Acrobat viewer will return is <i>PSNAME_SIZE</i> (see <i>PDExpT.h</i>).</p> |
| <b>Return Value</b>    | The number of characters in the font name. If the font name is too long to fit into <i>buffer</i> , <i>bufSize</i> - 1 bytes are copied into <i>buffer</i> , and <i>buffer</i> is <i>NULL</i> -terminated.                                                                                                                               |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                     |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                         |
| <b>Related Methods</b> | <a href="#">PDDocEnumFonts</a>                                                                                                                                                                                                                                                                                                           |
| <b>Example</b>         | <pre>PDWord pdWord; PDFont font; PDStyle style = PDWordGetNthCharStyle(pdWF,     pdWord, 0); size = PDStyleGetFontSize(style); PDFontGetName(font, mbuf, sizeof(mbuf)); if(size == fixedTen &amp;&amp; !strstr(mbuf,     "Italic")) {     /* process the 10 pt. Italic */     ... }</pre>                                                |

**Availability**

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFFontGetSubtype

[ASAtom](#) PDFFontGetSubtype ([PDFFont](#) font);

|                        |                                                                                                                                                                                                      |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets a font's subtype.                                                                                                                                                                               |
| <b>Parameters</b>      | <p><i>font</i></p> <p>The font whose subtype is obtained.</p>                                                                                                                                        |
| <b>Return Value</b>    | <p><i>subtype</i></p> <p>The font's subtype. The <i>ASAtom</i> returned can be converted to a string using <a href="#">ASAtomGetCount</a>.<br/>Must be one of the <a href="#">Font Subtypes</a>.</p> |
| <b>Exceptions</b>      | None                                                                                                                                                                                                 |
| <b>Notifications</b>   | None                                                                                                                                                                                                 |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                     |
| <b>Related Methods</b> | <a href="#">PDDocEnumFonts</a>                                                                                                                                                                       |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                             |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontGetWidths

```
void PDFontGetWidths (PDFont font, ASInt16* widths);
```

### Description

Gets the advance width of every glyph in a font. The advance width is the amount by which the current point advances when the glyph is drawn. The advance width may not correspond to the visible width of the glyph (for example, a glyph representing an accent mark might have an advance width of zero so that characters can be drawn under it). For this reason, the advance width cannot be used to determine the glyphs' bounding boxes.

For non-Roman character set viewers, this method gets the width for a single byte range (0 through 255).

### Parameters

*font*

The font whose glyph advance widths are obtained.

*widths*

*(Filled by the method)* An array of glyph advance widths, measured in character space units. Unencoded code points will have a width of zero. For non-Roman character set viewers, an array for a single byte range (0 through 255).

### Return Value

None

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDFontGetMetrics](#)  
[PDFontSetMetrics](#)  
[PDFontXlateWidths](#)

## Example

```
PDWordFinder wObj;
PDWord pdWord;
PDStyle style;
PDFont font;
PDFontMetrics fMetrics;
ASInt16 widths[256];

/* Get font width info */
style = PDWordGetNthCharStyle(wObj, pdWord,
 0);
font = PDStyleGetFont(style);
PDFontGetWidths(font, widths);
```

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontIsEmbedded

```
ASBool PDFontIsEmbedded (PDFont font);
```

**Description** Tests whether or not the specified font is embedded in the PDF file (that is, the font is stored as a font file, which is a stream embedded in the PDF file). Only Type 1 and TrueType fonts can be embedded.

**Parameters** *font*  
The font to test.

**Return Value** *true* if the font is embedded in the file, *false* otherwise.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocEnumFonts](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontSetMetrics

```
void PDFontSetMetrics (PDFont font, PDFontMetricsP metricsP,
 os_size_t sizeMetrics);
```

### Description

Sets a font's metrics, which provide the information needed to create a substitute multiple master font when the original font is unavailable. See Section 7.10 in the [Portable Document Format Reference Manual](#) for a discussion of font descriptors. This method can only be used on Type 1, multiple master Type 1, and TrueType fonts; it cannot be used on Type 3 fonts.

### Parameters

*font*

The font whose metrics are being set.

*metricsP*

Pointer to a [PDFontMetrics](#) structure containing the font's metrics.

*sizeMetrics*

Must be *sizeof(PDFontMetrics)*.

### Return Value

None

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDFontGetMetrics](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | —         | —       |





## PDFontXlateString

```
ASBool PDFontXlateString (PDFont font, ASUns8* inP,
 ASUns8* outP, ASInt32 len);
```

### Description

Translates a string into host encoding. If any characters cannot be represented in host encoding, they are replaced with space characters. If no *XlateTable* exists in the font, the function will return *false*, and *outP* will not be written.

For non-Roman character set viewers, it is *not* appropriate to call this method. Instead call one of the following: [PDFontXlateToHost](#), [PDFontXlateToUCS](#), [PDXlateToHostEx](#), or [PDXlateToPDFDocEncEx](#).

### Parameters

*font*

The font (and hence, encoding) that *inP* uses.

*inP*

The string to translate.

*outP*

(Filled by the method) The translated string. *outP* may point to the same buffer as *inP*, to allow in-place translation.

*len*

The length of *inP* and *outP*.

### Return Value

*true* if an *XlateTable* exists in the font, *false* otherwise. If no *XlateTable* exists in the font, *outP* will not be written.

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDXlateToHostEx](#)  
[PDXlateToPDFDocEncEx](#)  
[PDFontAcquireXlateTable](#)  
[PDFontXlateToHost](#)  
[PDFontXlateToUCS](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontXlateTableRelease

```
void PDFontXlateTableRelease (ASInt16* table);
```

|                        |                                                                                                                                                                                                                                                                                                       |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Decrements the specified font's <i>XlateTable</i> reference count. The <i>XlateTable</i> is a 256-entry table that maps characters from their encoding in the PDF file to host encoding. If a character cannot be mapped to host encoding, then the table entry will (for that character) contain -1. |
| <b>Parameters</b>      | <p><i>table</i></p> <p>The <i>XlateTable</i> to release.</p>                                                                                                                                                                                                                                          |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                  |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                  |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                  |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                      |
| <b>Related Methods</b> | <a href="#">PDFontAcquireXlateTable</a><br><a href="#">PDFontXlateString</a>                                                                                                                                                                                                                          |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                              |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDFontXlateToHost

```
ASInt32 PDFontXlateToHost (PDFont fontP, ASUns8* inP,
 ASInt32 inLen, ASUns8* outP, ASInt32 outLen);
```

### Description

Translates a string from the *PDFont*'s encoding to host encoding. This is useful for converting the text from a [PDWord](#) into host encoding. In the same way that [PDXlateToHostEx](#) converts text from bookmark titles to host encoding, *PDFontXlateToHost* converts text from a page contents stream to host encoding. Use [PDFontXlateToUCS](#) to translate from the *PDFont*'s encoding to Unicode.

Non-Roman fonts, such as PostScript composite fonts, can be encoded in different ways, such as SHIFT-JIS, RKSJ, and so on. To use *PDFontXlateToHost*, the caller doesn't need to know which encoding they're converting from, since that information is contained in the *PDFont*.

Host encoding is a platform-dependent encoding for the host machine. For non-UNIX Roman systems, it is MacRomanEncoding in Mac OS and WinAnsiEncoding in Windows. In UNIX (except HP-UX) Roman systems, it is ISO8859-1 (ISO Latin-1); for HP-UX, it is HP-ROMAN8. See Appendix C in the [Portable Document Format Reference Manual](#) for descriptions of MacRomanEncoding and WinAnsiEncoding. For non-Roman systems, the host encoding may be a variety of encodings, which are defined by a CMap (character map). See Section 7.9 in the [Portable Document Format Reference Manual](#) for a list of predefined CMaps. Use [PDGetHostEncoding](#) to determine if a system's host encoding is Roman or not.

### Parameters

*fontP*

The font used in the input string *inP*.

*inP*

Pointer to the string to translate.

*inLen*

Length of the *inP* buffer, in bytes.

*outP*

(Filled by the method) Pointer to the translated string.

*outLen*

The length of the *outP* buffer, in bytes.

**Return Value** Number of bytes in the translated string *outP*.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDFontXlateToUCS](#)  
[PDGetHostEncoding](#)  
[PDXlateToHostEx](#)  
[PDXlateToPDFDocEncEx](#)

**Example**

```
ASInt16 newlen;
ASUns8 stackBuffer[1024];
ASUns8* buffer = stackBuffer;

newlen = PDFontXlateToHost(font, string,
 stringlen, buffer,
 sizeof(stackBuffer) - 1);

if (newlen == sizeof(stackBuffer) - 1) {
 newlen = PDFontXlateToHost(font, string,
 stringlen, NULL, 0);
 buffer = ASmalloc(newlen + 1);
 newlen = PDFontXlateToHost(font, string,
 stringlen, buffer, newlen);
}

buffer[newlen] = '\0';
...
if (buffer != stackBuffer)
 ASfree(buffer);
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontXlateToUCS

```
ASInt32 PDFontXlateToUCS (PDFont fontP, ASUns8* inP,
 ASInt32 inLen, ASUns8* outP, ASInt32 outLen);
```

### Description

Translates a string from whatever encoding the *PDFont* uses to Unicode encoding. This is useful for converting the text from a *PDWord* into Unicode. Use *PDFontXlateToHost* to translate from the *PDFont*'s encoding to host encoding.

Non-Roman fonts, like PostScript composite fonts, can be encoded in different ways, such as SHIFT-JIS, RKSJ, and so on. The caller doesn't need to know which encoding they're converting from, since that information is contained in the *PDFont*.

For Roman systems, a font encoding describes a font's character encoding—the mapping between numeric character codes and character names. It may be MacRomanEncoding in Mac OS, ISO8859-1 (ISO Latin-1) on UNIX systems, WinAnsiEncoding in Windows, or some other encoding. See Section 7.8 in the [Portable Document Format Reference Manual](#) for information on font encodings. See Appendix C in the [Portable Document Format Reference Manual](#) for descriptions of MacRomanEncoding and WinAnsiEncoding.

For non-Roman systems, the font encoding may be a variety of encodings, which are defined by a CMap (character map). See Section 7.9 in the [Portable Document Format Reference Manual](#) for a list of predefined CMaps.

### Parameters

*fontP*

The font of the input string *inP*.

*inP*

Pointer to the string to translate.

*inLen*

Length of the *inP* buffer, in bytes.

*outP*

(Filled by the method) Pointer to the translated string. If *NULL*, the method returns the size of the translated string.



*outLen*

The length of the *outP* buffer, in bytes. If 0, the method returns the size of the translated string.

**Return Value** Number of bytes in the translated string in *outP*.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDFontXlateToHost](#)  
[PDGetHostEncoding](#)  
[PDXlateToHostEx](#)  
[PDXlateToPDFDocEncEx](#)

**Example**

```
ASInt32 newlen;
ASUns8 stackBuffer[1024];
ASUns8* buffer = stackBuffer;

newlen = PDFontXlateToUCS(font, string,
 stringlen, buffer,
 sizeof(stackBuffer) - 2);

if (newlen == sizeof(stackBuffer) - 2) {
 newlen = PDFontXlateToUCS(font, string,
 stringlen, NULL, 0);
 buffer = ASmalloc(newlen + 2);
 newlen = PDFontXlateToUCS(font, string,
 stringlen, buffer, newlen);
}

buffer[newlen] = '\0';
buffer[newlen+1] = '\0';
...
if (buffer != stackBuffer)
 ASfree(buffer);
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFontXlateWidths

```
void PDFontXlateWidths (PDFont font, ASInt16* inP,
 ASInt16* outP);
```

### Description

Translates an array of 256 glyph advance widths (obtained from [PDFontGetWidths](#)) from their order in the PDF file into host encoding order. If the widths are already in host encoding order, the widths are merely copied. All unencoded code points are given a width of zero.

For non-Roman character set viewers, it is *not* appropriate to call this method.

### Parameters

*font*

The font whose glyph widths are translated.

*inP*

Array of glyph advance widths to rearrange.

*outP*

(Filled by the method) Rearranged array of glyph advance widths.

### Return Value

None

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDFontGetWidths](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |



## PDFForm

## PDFFormEnumPaintProc

```
void PDFFormEnumPaintProc (PDXObject obj,
 PDGraphicEnumMonitor mon, void* clientData);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                     |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Enumerates a form's drawing operations.                                                                                                                                                                                                                                                                                                                             |
| <b>Parameters</b>      | <i>obj</i><br><br>The form whose drawing operations are enumerated.<br><br><i>mon</i><br><br>Structure containing user-supplied callbacks that are called for each drawing operator on a page. Enumeration ends if any procedure returns <i>false</i> .<br><br><i>clientData</i><br><br>Pointer to user-supplied data to pass to <i>mon</i> each time it is called. |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                    |
| <b>Related Methods</b> | <a href="#">PDFFormEnumResources</a>                                                                                                                                                                                                                                                                                                                                |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                            |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFFormEnumResources

```
void PDFFormEnumResources (PDXObject obj,
 PDResourceEnumMonitor mon, void* clientData);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                      |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Enumerates the resources used by a form.                                                                                                                                                                                                                                                                                                                             |
| <b>Parameters</b>      | <p><i>obj</i></p> <p>The form whose resources are enumerated.</p> <p><i>mon</i></p> <p>Structure containing user-supplied callbacks that are called for each of the form's resources.</p> <p>Enumeration ends if any procedure returns <i>false</i>.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>mon</i> each time it is called.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                     |
| <b>Related Methods</b> | <a href="#">PDFFormEnumPaintProc</a>                                                                                                                                                                                                                                                                                                                                 |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                             |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFFormGetBBox

```
void PDFFormGetBBox (PDXObject obj, ASFixedRect* bboxP);
```

|                        |                                                                                                                                                                                                                     |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets a form's bounding box.                                                                                                                                                                                         |
| <b>Parameters</b>      | <p><i>obj</i></p> <p>The form whose bounding box is obtained.</p> <p><i>bboxP</i></p> <p>(Filled by the method) Pointer to a rectangle containing the form's bounding box, specified in user space coordinates.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                |
| <b>Notifications</b>   | None                                                                                                                                                                                                                |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                    |
| <b>Related Methods</b> | None                                                                                                                                                                                                                |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                            |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |



## PDFFormGetFormType

```
ASInt32 PDFFormGetFormType (PDXObject obj);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the value of a form's <i>FormType</i> attribute.                                                    |
| <b>Parameters</b>      | <i>obj</i><br>Form whose type is obtained.                                                               |
| <b>Return Value</b>    | Form type (value of the PDF <i>FormType</i> key). This value is 1 for PDF 1.0, 1.1, and 1.2.             |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | None                                                                                                     |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher. |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDFFormGetMatrix

```
void PDFFormGetMatrix (PDXObject obj, ASFixedMatrix* matrixP);
```

|                        |                                                                                                                                                                                                                                                                                                                                                       |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the specified form's transformation matrix.                                                                                                                                                                                                                                                                                                      |
| <b>Parameters</b>      | <p><i>obj</i></p> <p>The form whose transformation matrix is obtained.</p> <p><i>matrix P</i></p> <p>(Filled by the method) Pointer to a matrix containing the form's transformation matrix, which specifies the transformation from form space to user space. See Section 3.10 in the <a href="#">Portable Document Format Reference Manual</a>.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                  |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                  |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                  |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                      |
| <b>Related Methods</b> | None                                                                                                                                                                                                                                                                                                                                                  |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                              |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDFFormGetXUIDCosObj

**CosObj** PDFFormGetXUIDCosObj (**PDXObject** obj);

|                        |                                                                                                                                                                            |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the array Cos object corresponding to a form's <i>XUID</i> . An <i>XUID</i> is an array of numbers that uniquely identify the form in order to allow it to be cached. |
| <b>Parameters</b>      | <i>obj</i><br>The form whose <i>XUID</i> is obtained.                                                                                                                      |
| <b>Return Value</b>    | The array Cos object for the form's <i>XUID</i> .                                                                                                                          |
| <b>Exceptions</b>      | None                                                                                                                                                                       |
| <b>Notifications</b>   | None                                                                                                                                                                       |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                           |
| <b>Related Methods</b> | None                                                                                                                                                                       |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                   |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

**PDGraphic**

## PDGraphicGetBBox

```
void PDGraphicGetBBox (PDGraphic obj, ASFixedRect* bboxP);
```

|                        |                                                                                                                                                                                                                                                                                                                     |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets a bounding box for the specified graphic object.                                                                                                                                                                                                                                                               |
| <b>Parameters</b>      | <p><i>obj</i></p> <p>The graphic object whose bounding box is obtained.</p> <p><i>bboxP</i></p> <p>(Filled by the method) If called during <a href="#">PDFormEnumPaintProc</a> or <a href="#">PDCharProcEnum</a>, the coordinates are specified in the object space (that is, relative to the object's matrix).</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                    |
| <b>Related Methods</b> | <a href="#">PDCharProcEnum</a><br><a href="#">PDFormEnumPaintProc</a>                                                                                                                                                                                                                                               |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                            |

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDGraphicGetCurrentMatrix

```
void PDGraphicGetCurrentMatrix (PDGraphic obj,
 ASFixedMatrix* matrix);
```

|                        |                                                                                                                                                                                                                      |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the current transformation matrix in effect for a graphic object; the matrix is relative to user space.                                                                                                         |
| <b>Parameters</b>      | <p><i>obj</i></p> <p>The graphic object for which transformation matrix is obtained.</p> <p><i>matrix</i></p> <p>(Filled by the method) Pointer to a matrix containing the transformation matrix for <i>obj</i>.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                 |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                 |
| <b>Notifications</b>   | None                                                                                                                                                                                                                 |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                     |
| <b>Related Methods</b> | None                                                                                                                                                                                                                 |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                             |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDGraphicGetState

```
void PDGraphicGetState (PDGraphic obj, PDGraphicStateP stateP,
 ASInt32 stateLen);
```

|                        |                                                                                                                                                                                                                                                                                             |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the graphics state associated with a graphic object. See Section 8.2 in the <a href="#">Portable Document Format Reference Manual</a> for a discussion of the graphics state parameters.                                                                                               |
| <b>Parameters</b>      | <p><i>obj</i></p> <p>The graphic object whose graphics state is obtained.</p> <p><i>stateP</i></p> <p>(Filled by the method) Pointer to a <a href="#">PDGraphicState</a> structure containing the graphics state.</p> <p><i>stateLen</i></p> <p>Must be <i>sizeof(PDGraphicsState)</i>.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                        |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                        |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                        |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                            |
| <b>Related Methods</b> | None                                                                                                                                                                                                                                                                                        |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                    |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## **PDImage**



## PDImageColorSpaceGetIndexLookup

```
void PDImageColorSpaceGetIndexLookup (PDXObject obj,
 ASUns8* data, ASInt32 dataLen);
```

**Description**

Gets the lookup table for an indexed color space. The table will contain the number of entries specified by the index size, and there will be 1 byte for each color component for each entry. The number of color components depends on the color space: gray→1, RGB→3, CMYK→4, Lab→3. For additional information on indexed color space, see Section 7.11.7 in the [Portable Document Format Reference Manual](#). There is also some useful discussion in the *PostScript Language Reference Manual, Second Edition* under indexed color spaces.

**Parameters**

*obj*

The image whose lookup table is obtained.

*data*

(Filled by the method) Array for returned color space information.

*dataLen*

Length of *data*, in bytes.

**Return Value**

None

**Exceptions**

None

**Notifications**

None

**Header File**

*PDCalls.h*

**Related Methods**

[PDInlineImageColorSpaceGetIndexLookup](#)

**Availability**

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Reader      | M, W, U   | M, W, U |

## PDImageGetAttrs

```
void PDImageGetAttrs (PDXObject obj, PDImageAttrsP attrsP,
 ASInt32 attrsLen);
```

|                        |                                                                                                                                                                                                                                                                              |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the attributes of an image (for example, Type, Subtype, Name, Width, Height, BitsPerComponent, ColorSpace, Decode, Interpolate, ImageMask).                                                                                                                             |
| <b>Parameters</b>      | <p><i>obj</i></p> <p>The image whose attributes are obtained.</p> <p><i>attrsP</i></p> <p>(Filled by the method) Pointer to a <a href="#">PDImageAttrs</a> structure containing the image attributes.</p> <p><i>attrsLen</i></p> <p>Must be <i>sizeof(PDImageAttrs)</i>.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                         |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                         |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                         |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                             |
| <b>Related Methods</b> | <a href="#">PDInlineImageGetAttrs</a>                                                                                                                                                                                                                                        |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                     |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDImageSelectAlternate

```
CosObj PDImageSelectAlternate (CosObj image, ASBool print,
 ASUns32 tickLimit, ASBool* cacheImageP, void* callData);
```

### Description

Selects which Alternate image to use. It can do one of three things:

- Return an existing Alternate
- Create an image *XObject* and return that
- Indicate that this *XObject* should be skipped

This method is called each time the Acrobat viewer draws an *XObject* image, regardless of whether or not the image *XObject* has an Alternates key.

You can replace this method with your own version, using [HFTReplaceEntry](#).

### Parameters

*image*

The Cos object for this image *XObject*'s base image.

*print*

*true* if printing, *false* if displaying.

*tickLimit*

Amount of time, in ticks, before the image selector must return control to the Acrobat viewer. This parameter is not relevant for image selectors that simply choose an existing alternate or create a new image *XObject* using Cos calls, but it is relevant for image selectors that create a new image *XObject* by calculating or reading data from a network or a slow device.

If *tickLimit* is zero, the image selector must not return control to the Acrobat viewer until the selector can provide the alternate image to use.

If *tickLimit* is non-zero, the image selector does not have to provide the image *XObject* within *tickLimit*, but it must raise *bytesNotReady* if it cannot. This returns control to the Acrobat viewer and informs it that the data is not ready. The Acrobat viewer will then call the image selector periodically until the image selector returns the selected image *XObject*.

## *cacheImageP*

If *true*, the image data returned to the Acrobat viewer will be cached for future use. If *false*, they will not. Pass *true* if you expect your image data to remain valid for at least several calls to your plug-in. Pass *false* if you expect your image data to change on the next call to your plug-in.

If you create and return a Cos stream object with an external file system and the stream's data will not always be the same, you must set *cacheImageP* to *false*. If you fail to do this, the Acrobat viewer may improperly use cached image data when it should not.

## *callData*

An opaque pointer containing data that must be passed in other image selector calls.

**Return Value** The image *XObject* to use. Returning a *NULL* Cos object (obtained using *CosNewNull*) tells the Acrobat viewer to skip this *XObject* entirely.

**Exceptions** [fileErrBytesNotReady](#)

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDImageSelGetGeoAttr](#)  
[PDImageSelGetDeviceAttr](#)  
[PDImageSelAdjustMatrix](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

## Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDImageSelAdjustMatrix

```
void PDImageSelAdjustMatrix (void* callData,
 ASFixedMatrix newUserMatrix);
```

### Description

Allows an image selector plug-in to change the region of the page occupied by an image. It must only be used by image selector plug-ins that return data for only the visible part of an image, to set the region of the page that the sub-image occupies. It must not be used otherwise.

This method only has an effect while displaying on screen. It does nothing when printing.

The matrix set by this call remains in effect only for the current image. The Acrobat viewer automatically replaces it after the image has been drawn.

### Parameters

*callData*

The value passed to the image selector as a parameter to [PDImageSelectAlternate](#).

*newUserMatrix*

The matrix that will replace the *imageToUserMatrix* (see [PDImageSelGetGeoAttr](#)). The *imageToDevMatrix* is automatically calculated from *newUserMatrix*.

### Return Value

None

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDImageSelGetGeoAttr](#)  
[PDImageSelGetDeviceAttr](#)  
[PDImageSelectAlternate](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

### Available in:

| Library API       | 1.0 | 4.0     |
|-------------------|-----|---------|
| Adobe PDF Library | —   | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDImageSelGetDeviceAttr

```
void PDImageSelGetDeviceAttr (void* callData,
 PDColorSpace* colorSpaceP, ASUns32* bitsPerPixelP,
 ASAAtom* deviceTypeP);
```

### Description

Gets device-related attributes for a particular image *XObject*. It must only be used from within an image selector plug-in, since it returns information that is only valid in that context. This method can be used by an image selector plug-in to obtain additional information to help the selector determine which alternate image to choose.

If an image is displayed on two devices simultaneously (for example, if the window containing the image is split across two monitors in a multi-monitor system), the values returned for *colorSpaceP* and *bitsPerPixelP* are each the maximum for the devices on which the image is currently displayed. For example, if the image is currently split across the following devices:

- 8-bit grayscale monitor
- 4-bit RGB color monitor

*colorSpaceP* would be **DeviceRGB**, because that is the “highest” colorspace on which the image is currently displayed.

*bitsPerPixelP* would be 8, because that is the highest bit depth on which the image is currently displayed.

### Parameters

*callData*

The value passed to the image selector as a parameter to [PDImageSelectAlternate](#).

*colorSpaceP*

Destination device’s colorspace. Must be one of the following:

*PDDeviceGray* — Grayscale device

*PDDeviceRGB* — RGB device

*PDDeviceCMYK* — CMYK device

If the device has some other color space, or its color space cannot be determined, **PDDeviceRGB** is returned.



## *bitsPerPixel*

Number of bits used for each pixel (that is, a device with 8 bits red, 8 bits green, and 8 bits blue) would have a *bitsPerPixel* of 24.

If *bitsPerPixel* has a value of 0 (instead of the more standard 1, 8, or 24) it means that we could not determine the number of bits per pixel on the output device.

## *deviceType*

Output device type. Must be one of the following:

*Display* — A display device such as a monitor

*PostScript* — A PostScript printer or PostScript file

*nonPostScriptPrinter* — a non-PostScript printer

**Return Value** None

**Header File** *PDCalls.h*

**Related Methods** [PDImageSelAdjustMatrix](#)  
[PDImageSelGetDeviceAttr](#)  
[PDImageSelectAlternate](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

## Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDImageSelGetGeoAttr

```
void PDImageSelGetGeoAttr (void* callData,
 ASFixedRect* updateBBoxP, ASFixedMatrix* imageToDefaultMatrixP,
 ASFixedMatrix* imageToDevMatrixP);
```

**Description** Requests geometry-related attributes of an image *XObject*. This method can be used by an image selector plug-in to obtain additional information to help the selector determine which alternate image to choose.

**Parameters** *callData*

The value passed to the image selector as a parameter to [PDImageSelectAlternate](#).

*updateBBoxP*

Rectangle bounding the region of the page to update. This is the intersection of the visible portion of the page, any update regions, and any clip paths that have been explicitly set in the PDF file. Its coordinates are specified in default user space units (that is, 72 units per inch).

*imageToDefaultMatrix*

Matrix specifying the transformation from image space (the space in which all images are 1x1 units) to default user space (the space with 72 units per inch). It contains sufficient information for the image selector plug-in to determine the image's size and location on the page. For a "normal page and image" (an "upright" image on a non-rotated page), the following is true:

imageToDefaultMatrixP ->a = image horizontal size in 1/72 of an inch units

imageToDefaultMatrixP ->b = 0

imageToDefaultMatrixP ->c = 0

imageToDefaultMatrixP ->d = image vertical size in 1/72 of an inch units

imageToDefaultMatrixP->h = left edge of image

imageToDefaultMatrixP->v = bottom edge of image

In other words, this matrix provides the image's height, width, and position on the page, all in units of points (compare to *imageToDefaultMatrixP*).

The intersection of the rectangle obtained by transforming a 1x1 unit rectangle (the image) through *imageToDeviceMatrixP* and the *updateBBoxP* rectangle, is the region of the image that will actually be drawn. This is the region of the image for which data is required.

## *imageToDevMatrixP*

Matrix specifying the transformation from image space (the space in which all images are 1x1 unit) to device space (the space in which one unit is one pixel). This matrix provides the image's height, width, and position on the page, all in pixels (compare to *imageToDefaultMatrixP*).

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDImageSelAdjustMatrix](#)  
[PDImageSelGetDeviceAttr](#)  
[PDImageSelectAlternate](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

## Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | —         | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | —         | M, W    |
| Reader            | —         | M, W, U |

**PDInlinelImage**

## PDInlineImageColorSpaceGetIndexLookup

```
void PDInlineImageColorSpaceGetIndexLookup (PDInlineImage image,
 ASUns8* data, ASInt32 dataLen);
```

**Description**

Gets the lookup table for an indexed color space. The table will contain the number of entries specified by the index size, and there will be 1 byte for each color component for each entry. The number of color components depends on the color space: gray→1, RGB→3, CMYK→4, Lab→3. For additional information on indexed color space, see Section 7.11.7 in the [Portable Document Format Reference Manual](#). There is also some useful discussion in the *PostScript Language Reference Manual, Second Edition* under indexed color spaces.

**Parameters**

*image*

The inline image whose lookup table is obtained.

*data*

(Filled by the method) The lookup table for *image*.

*dataLen*

Length of *data*, in bytes.

**Return Value**

None

**Exceptions**

None

**Notifications**

None

**Header File**

*PDCalls.h*

**Related Methods**

[PDImageColorSpaceGetIndexLookup](#)

**Availability**

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Reader      | M, W, U   | M, W, U |

## PDInlineImageGetAttrs

```
void PDInlineImageGetAttrs (PDInlineImage obj,
 PDImageAttrsP attrsP, ASInt32 attrsLen);
```

### Description

Gets an inline image's attributes.

*Note: The attribute for a color space is a CosObj. Cos objects that are the result of parsing inline dictionaries in the PDF page contents are a special class of Cos objects. You should never depend on these objects lasting the lifetime of the document. You should extract the information you need from the object immediately and refer to it no further in your code.*

### Parameters

*obj*

The inline image whose attributes are obtained.

*attrsP*

(Filled by the method) Pointer to a [PDImageAttrs](#) structure containing the image attributes.

*Note: This structure contains a Cos object that is subject to the warning above.*

*attrsLen*

Must be `sizeof(PDImageAttrs)`.

### Return Value

None

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDImageGetAttrs](#)  
[PDInlineImageGetData](#)

Example

```
PDImageAttrs iattrs;
ASUns8* bitsData;

PDInlineImageGetAttrs(obj,&iattrs,
 sizeof(PDImageAttrs));

if(NULL == (bitsData =
 (ASUns8 *)ASmalloc(iattrs.dataLen))){
 AVAlertNote("Serious Memory Allocation
Error");
 return false;
}

PDInlineImageGetData(obj,bitsData,
 iattrs.dataLen);
```

Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDInlineImageGetData

```
void PDInlineImageGetData (PDInlineImage obj, ASUns8* data,
 ASInt32 dataLen);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                           |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the image data for an inline image.                                                                                                                                                                                                                                                                                                                                                  |
| <b>Parameters</b>      | <p><i>obj</i></p> <p>The inline image whose data is obtained.</p> <p><i>data</i></p> <p>(Filled by the method) A buffer into which the image data will be placed.</p> <p><i>dataLen</i></p> <p>Number of bytes that <i>data</i> can hold. Must be large enough to hold the entire inline image. Use <a href="#">PDInlineImageGetAttrs</a> to determine how much data is in the image.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Exceptions</b>      | Raises <a href="#">genErrBadParm</a> if <i>dataLen</i> < the amount of data in the image.                                                                                                                                                                                                                                                                                                 |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Related Methods</b> | <a href="#">PDInlineImageGetAttrs</a>                                                                                                                                                                                                                                                                                                                                                     |
| <b>Example</b>         | <pre>PDImageAttrs iattrs; ASUns8* bitsData;  PDInlineImageGetAttrs(obj,&amp;iattrs,     sizeof(PDImageAttrs));  if(NULL == (bitsData =     (ASUns8 *)ASmalloc(iattrs.dataLen)) ){     AVAlertNote("Serious Memory Allocation Error");     return false; }  PDInlineImageGetData(obj,bitsData,     iattrs.dataLen);</pre>                                                                  |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                                                  |

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

**PDLinkAnnot**

## PDLinkAnnotGetAction

[PDAction](#) PDLinkAnnotGetAction ([PDLinkAnnot](#) aLinkAnnot);

|                        |                                                                                                                                                                                                                                   |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets a link annotation's action. After you obtain the action, you can execute it with <a href="#">AVDocPerformAction</a> .                                                                                                        |
| <b>Parameters</b>      | <i>aLinkAnnot</i><br>The link annotation whose action is obtained.                                                                                                                                                                |
| <b>Return Value</b>    | The link annotation's action.                                                                                                                                                                                                     |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                              |
| <b>Notifications</b>   | None                                                                                                                                                                                                                              |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                  |
| <b>Related Methods</b> | <a href="#">AVDocPerformAction</a><br><a href="#">PDLinkAnnotSetAction</a><br><a href="#">PDActionNew</a><br><a href="#">PDActionNewFromDest</a><br><a href="#">PDActionNewFromFileSpec</a><br><a href="#">PDActionFromCosObj</a> |

**Example**

```
PDLinkAnnot pdAnnot;
PDAction linkAction;
PDViewDestination viewDest;
ASInt32 page;
ASAtom fit;
ASFixedRect destRect;
ASFixed zoom;

linkAction = PDLinkAnnotGetAction(pdAnnot);
viewDest = PDActionGetDest(oldAction);
PDViewDestGetAttr(viewDest, &page, &fit,
 &destRect, &zoom);
AVDocPerformAction(AVAppGetActiveDoc(),
 linkAction);
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDLinkAnnotGetBorder

```
void PDLinkAnnotGetBorder (PDLinkAnnot aLinkAnnot,
 PDLinkAnnotBorder* border);
```

|                        |                                                                                                                                                                                                                                                                                                  |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the border of a link annotation.                                                                                                                                                                                                                                                            |
| <b>Parameters</b>      | <p><i>aLinkAnnot</i></p> <p>The link annotation whose border is obtained.</p> <p><i>border</i></p> <p>(Filled by the method) Pointer to a structure containing the link border. Link corner radii are ignored by the Acrobat viewers.</p>                                                        |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                             |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                             |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                             |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                 |
| <b>Related Methods</b> | <a href="#">PDLinkAnnotSetBorder</a>                                                                                                                                                                                                                                                             |
| <b>Example</b>         | <pre>pdAnnot = PDPageAddNewAnnot(conPage,     PDPageGetNumAnnots(conPage)-1,     ASAtomFromString("Link"), &amp;dstRect); linkBorder.hCornerRadius = 16; linkBorder.vCornerRadius = 16; linkBorder.dashArrayLen = 0; linkBorder.width = 1; PDLinkAnnotSetBorder(pdAnnot, &amp;linkBorder);</pre> |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                         |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDLinkAnnotSetAction

```
void PDLinkAnnotSetAction (PDLinkAnnot aLinkAnnot,
 PDAction action);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Sets a link annotation's action.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Parameters</b>      | <p><i>aLinkAnnot</i></p> <p>The link annotation whose action is set.</p> <p><i>action</i></p> <p>New action for the link annotation.</p>                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Notifications</b>   | <a href="#">PDAnnotWillChange</a><br><a href="#">PDAnnotDidChange</a>                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Related Methods</b> | <a href="#">PDLinkAnnotGetAction</a><br><a href="#">PDActionNew</a><br><a href="#">PDActionNewFromDest</a><br><a href="#">PDActionNewFromFileSpec</a><br><a href="#">PDActionFromCosObj</a>                                                                                                                                                                                                                                                                                                                       |
| <b>Example</b>         | <pre>PDLinkAnnot pdAnnot;<br/>PDAction oldAction;<br/>PDViewDestination viewDest;<br/>ASInt32 page;<br/>ASAtom fit;<br/>ASFixedRect destRect;<br/>ASFixed zoom;<br/>PDPage tmp;<br/><br/>oldAction = PDLinkAnnotGetAction(pdAnnot);<br/>viewDest = PDActionGetDest(oldAction);<br/>PDViewDestGetAttr(viewDest, &amp;page, &amp;fit,<br/>    &amp;destRect, &amp;zoom);<br/>PDActionDestroy(oldAction);<br/>tmp = PDDocAcquirePage(pdDoc, page);<br/>PDAnnotGetTitle(pdAnnot, title,<br/>    sizeof(title));</pre> |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                                                                                                                                                                          |

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |



## PDLinkAnnotSetBorder

```
void PDLinkAnnotSetBorder (PDLinkAnnot aLinkAnnot,
 const PDLinkAnnotBorder* border);
```

|                        |                                                                                                                                                                                                                                                                                                  |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Sets a link annotation's border.                                                                                                                                                                                                                                                                 |
| <b>Parameters</b>      | <p><i>aLinkAnnot</i></p> <p>The link annotation whose border is set.</p> <p><i>border</i></p> <p>Pointer to a structure containing the link border. Link corner radii are ignored by the Acrobat viewers.</p>                                                                                    |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                             |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                             |
| <b>Notifications</b>   | <a href="#">PDAnnotWillChange</a><br><a href="#">PDAnnotDidChange</a>                                                                                                                                                                                                                            |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                 |
| <b>Related Methods</b> | <a href="#">PDLinkAnnotGetBorder</a>                                                                                                                                                                                                                                                             |
| <b>Example</b>         | <pre>pdAnnot = PDPageAddNewAnnot(conPage,     PDPageGetNumAnnots(conPage)-1,     ASAtomFromString("Link"), &amp;dstRect); linkBorder.hCornerRadius = 16; linkBorder.vCornerRadius = 16; linkBorder.dashArrayLen = 0; linkBorder.width = 1; PDLinkAnnotSetBorder(pdAnnot, &amp;linkBorder);</pre> |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                         |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | —         | —       |



## **PDNameTree**

## PDNameTreeEnum

```
void PDNameTreeEnum (PDNameTree theTree, CosObjEnumProc proc,
void* clientData);
```

|                        |                                                                                                                                                                                                                                                                                                                                   |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Enumerates the entries in the tree.                                                                                                                                                                                                                                                                                               |
| <b>Parameters</b>      | <p><i>theTree</i></p> <p>A name tree.</p> <p><i>proc</i></p> <p>A procedure to call once for each name destination pair in <i>theTree</i>.</p> <p><i>clientData</i></p> <p>Data used by the enumeration procedure. <i>clientData</i> is passed to the enumeration procedure <i>proc</i> each time a name tree is encountered.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                              |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                              |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                              |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                  |
| <b>Related Methods</b> | <a href="#">PDNameTreeGet</a><br><a href="#">PDNameTreePut</a><br><a href="#">PDNameTreeRemove</a>                                                                                                                                                                                                                                |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                                                          |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDNameTreeEqual

```
ASBool PDNameTreeEqual (PDNameTree tree1, PDNameTree tree2);
```

**Description** Compares two name trees to determine if they are the same object.

**Parameters**

*tree1*  
A name tree.

*tree2*  
Another name tree.

**Return Value** *true* if the two name trees are equivalent, *false* otherwise.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDNameTreesValid](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

**Available in:**

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDNameTreeFromCosObj

[PDNameTree](#) PDNameTreeFromCosObj ([CosObj](#) obj);

**Description** Creates a type cast of the *cosObj* to a name tree.

**Parameters** *obj*  
The *CosObj* for which a *PDNameTree* representation is desired.

**Return Value** A *PDNameTree* representation of *obj*.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDNameTreeNew](#)  
[PDNameTreeGetCosObj](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDNameTreeGet

```
ASBool PDNameTreeGet (PDNameTree theTree, const char* name,
 ASInt32 nameLen, CosObj* value);
```

|                        |                                                                                                                                                                                                                                                                                                                  |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Retrieves an object from the name tree.                                                                                                                                                                                                                                                                          |
| <b>Parameters</b>      | <p><i>theTree</i><br/>The <i>PDNameTree</i> requested.</p> <p><i>name</i><br/>The name of the object within <i>theTree</i> to desire.</p> <p><i>nameLen</i><br/>The length of <i>name</i>.</p> <p><i>value</i><br/>(Filled by the method) The Cos object corresponding to <i>name</i> within <i>theTree</i>.</p> |
| <b>Return Value</b>    | <i>true</i> if the object was retrieved, <i>false</i> if no object with this name exists.                                                                                                                                                                                                                        |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                             |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                             |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                 |
| <b>Related Methods</b> | <a href="#">PDNameTreeEnum</a><br><a href="#">PDNameTreePut</a><br><a href="#">PDNameTreeRemove</a>                                                                                                                                                                                                              |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                                         |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDNameTreeGetCosObj

```
CosObj PDNameTreeGetCosObj (PDNameTree theTree);
```

**Description** Creates a type cast of the name tree to a *cosObj*.

**Parameters** *theTree*  
The *PDNameTree* for which a *CosObj* representation is desired.

**Return Value** A *CosObj* representation of *theTree*.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDNameTreeNew](#)  
[PDNameTreeFromCosObj](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |



## PDNameTreeIsValid

```
ASBool PDNameTreeIsValid (PDNameTree theTree);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Validates whether a <i>PDNameTree</i> is a <i>CosDict</i> Cos object or not                              |
| <b>Parameters</b>      | <p><i>theTree</i></p> <p>The <i>PDNameTree</i> whose validity is desired.</p>                            |
| <b>Return Value</b>    | <i>true</i> if the name tree is a <i>CosDict</i> , <i>false</i> otherwise.                               |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDNameTreeEqual</a>                                                                          |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher. |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | —         | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | —         | M, W    |
| Reader            | —         | M, W, U |

## PDNameTreeLookup

```
CosObj PDNameTreeLookup (CosObj nameTree, char* name,
 ASInt32 nameLen);
```

### Description

Given a name tree, such as the Dests tree in the **Names** dictionary, and a string, find the *CosObj* in the tree that matches the string.

See Section 7.19 in the [Portable Document Format Reference Manual](#) for more information on name trees.

### Parameters

*nameTree*

The name tree in which to search.

*name*

The name to search for.

The name tree uses Cos-style strings, which may use Unicode encoding, and these may contain bytes with zeroes in them (high bytes of ASCII characters).

*Note: name is not a C-style string. Cos string objects can contain NULL chars. Standard C string-handling functions may not work as expected.*

*nameLen*

Length of *name*, in bytes.

### Return Value

The Cos object associated with the specified name, which is the array element following the name.

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDNameTreeGet](#)  
[PDNameTreePut](#)  
[PDNameTreeRemove](#)

## Example

```
CosObj result = CosNewNull();
CosObj nameTree, name;
char* pName, *p;
ASInt32 nameLen;

/* Assume nameTree and name have appropriate
 values */
p = CosStringValue(name, &nameLen);
pName = (char *) ASmalloc(nameLen);
p = CosStringValue(name, &nameLen); /*
 acquire again in case malloc caused a
 flush */
memcpy(pName, p, nameLen);

result = PDNameTreeLookup(nameTree, pName,
 nameLen);
```

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

## Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDNameTreeNew

[PDNameTree](#) PDNameTreeNew ( [PDDoc](#) pdDoc ) ;

|                        |                                                                                                                                                                                                                                                                                               |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Creates a new name tree in the document.                                                                                                                                                                                                                                                      |
| <b>Parameters</b>      | <p><i>pdDoc</i></p> <p>The document for which a new name tree is desired.</p>                                                                                                                                                                                                                 |
| <b>Return Value</b>    | <p>The newly-created name tree or a <i>NULL CosObj</i> if Acrobat is unable to create a <i>PDNameTree</i> for the document specified by <i>pdDoc</i>.</p> <p><a href="#">PDNameTreeValid</a> should be called to determine if the number tree returned by <i>PDNameTreeNew</i> is usable.</p> |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                          |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                          |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                              |
| <b>Related Methods</b> | <a href="#">PDNameTreeFromCosObj</a><br><a href="#">PDNameTreeGetCosObj</a><br><a href="#">PDNameTreeValid</a>                                                                                                                                                                                |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                      |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## PDNameTreePut

```
void PDNameTreePut (PDNameTree theTree, CosObj key,
 CosObj value);
```

**Description** Puts a new entry in the name tree. If an entry with this name is already in the tree, it is replaced.

**Parameters**

*theTree*  
The name tree for which a new entry is added

*key*  
A *CosString* (not a char \*, length). This allows the use of an existing indirect object for the key rather than forcing the creation of a new object.

*value*  
The value associated with *key*.

**Return Value** None

**Exceptions** None

**Notifications** [PDNameTreeNameAdded](#)

**Header File** *PDCalls.h*

**Related Methods** [PDNameTreeEnum](#)  
[PDNameTreeGet](#)  
[PDNameTreeRemove](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## PDNameTreeRemove

```
void PDNameTreeRemove (PDNameTree theTree, const char* key,
 ASInt32 keyLen);
```

|                        |                                                                                                                                                                                                |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Removes the specified object from the tree. Does nothing if no object with that name exists.                                                                                                   |
| <b>Parameters</b>      | <p><i>theTree</i></p> <p>The name tree from which an entry is removed.</p> <p><i>key</i></p> <p>The name of the entry to remove.</p> <p><i>keyLen</i></p> <p>Length of key name, in bytes.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                           |
| <b>Exceptions</b>      | None                                                                                                                                                                                           |
| <b>Notifications</b>   | <a href="#">PDNameTreeNameRemoved</a>                                                                                                                                                          |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                               |
| <b>Related Methods</b> | <a href="#">PDNameTreeEnum</a><br><a href="#">PDNameTreeGet</a><br><a href="#">PDNameTreePut</a>                                                                                               |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                       |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## **PDNumTree**

## PDNumTreeEnum

```
void PDNumTreeEnum (PDNumTree theTree, CosObjEnumProc proc,
void* clientData);
```

|                        |                                                                                                                                                                                                                                                                                                                                         |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Enumerates the entries in the tree.                                                                                                                                                                                                                                                                                                     |
| <b>Parameters</b>      | <p><i>theTree</i></p> <p>A number tree.</p> <p><i>proc</i></p> <p>A procedure to call once for each number destination pair in <i>theTree</i>.</p> <p><i>clientData</i></p> <p>Data used by the enumeration procedure. <i>clientData</i> is passed to the enumeration procedure <i>proc</i> each time a number tree is encountered.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                    |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                    |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                    |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                        |
| <b>Related Methods</b> | <a href="#">PDNumTreeGet</a><br><a href="#">PDNumTreePut</a><br><a href="#">PDNumTreeRemove</a>                                                                                                                                                                                                                                         |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                                                                |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |



## PDNumTreeEqual

```
ASBool PDNumTreeEqual (PDNumTree tree1, PDNumTree tree2);
```

**Description** Compares two number trees to determine if they are the same object.

**Parameters**

*tree1*  
A number tree.

*tree2*  
Another number tree.

**Return Value** *true* if the two number trees are equivalent, *false* otherwise.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDNumTreesValid](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

**Available in:**

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | —         | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | —         | M, W    |
| Reader            | —         | M, W, U |

## PDNumTreeFromCosObj

[PDNumTree](#) PDNumTreeFromCosObj ( [CosObj](#) obj );

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Creates a type cast of the <i>cosObj</i> to a number tree.                                               |
| <b>Parameters</b>      | <p><i>obj</i></p> <p>The <i>CosObj</i> for which a <i>PDNumTree</i> representation is desired.</p>       |
| <b>Return Value</b>    | A <i>PDNumTree</i> representation of <i>obj</i> .                                                        |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDNumTreeNew</a><br><a href="#">PDNumTreeGetCosObj</a>                                       |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher. |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDNumTreeGet

```
ASBool PDNumTreeGet (PDNumTree theTree, ASInt32 key,
 CosObj* value);
```

|                        |                                                                                                                                                                                                           |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Retrieves an object from the number tree.                                                                                                                                                                 |
| <b>Parameters</b>      | <p><i>theTree</i><br/>The <i>PDNumTree</i> requested.</p> <p><i>key</i><br/>The number of the entry to retrieve.</p> <p><i>value</i><br/>(Filled by the method) The value associated with <i>key</i>.</p> |
| <b>Return Value</b>    | <i>true</i> if the object was retrieved, <i>false</i> if no object with this number exists.                                                                                                               |
| <b>Exceptions</b>      | None                                                                                                                                                                                                      |
| <b>Notifications</b>   | None                                                                                                                                                                                                      |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                          |
| <b>Related Methods</b> | <a href="#">PDNumTreeEnum</a><br><a href="#">PDNumTreePut</a><br><a href="#">PDNumTreeRemove</a>                                                                                                          |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                  |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDNumTreeGetCosObj

[CosObj](#) PDNumTreeGetCosObj ([PDNumTree](#) theTree);

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Creates a type cast of the number tree to a <i>cosObj</i> .                                              |
| <b>Parameters</b>      | <i>theTree</i><br>The <i>PDNumTree</i> for which a <i>CosObj</i> representation is desired.              |
| <b>Return Value</b>    | A <i>CosObj</i> representation of <i>theTree</i> .                                                       |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDNumTreeNew</a><br><a href="#">PDNumTreeFromCosObj</a>                                      |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher. |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDNumTreeIsValid

```
ASBool PDNumTreeIsValid (PDNumTree theTree);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Validates whether a <i>PDNumTree</i> is a <i>CosDict</i> Cos object or not                               |
| <b>Parameters</b>      | <i>theTree</i><br>The <i>PDNumTree</i> whose validity is desired.                                        |
| <b>Return Value</b>    | <i>true</i> if the number tree is a <i>CosDict</i> , <i>false</i> otherwise.                             |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDNumTreeEqual</a>                                                                           |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher. |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | —         | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | —         | M, W    |
| Reader            | —         | M, W, U |

## PDNumTreeNew

```
PDNumTree PDNumTreeNew (PDDoc pdDoc) ;
```

|                        |                                                                                                                                                                                                                                                                                               |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Creates a new number tree in the document.                                                                                                                                                                                                                                                    |
| <b>Parameters</b>      | <p><i>pdDoc</i></p> <p>The document for which a new number tree is desired.</p>                                                                                                                                                                                                               |
| <b>Return Value</b>    | <p>The newly-created number tree or a <i>NULL CosObj</i> if Acrobat is unable to create a <i>PDNumTree</i> for the document specified by <i>pdDoc</i>.</p> <p><a href="#">PDNumTreesValid</a> should be called to determine if the number tree returned by <i>PDNumTreeNew</i> is usable.</p> |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                          |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                          |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                              |
| <b>Related Methods</b> | <a href="#">PDNumTreeFromCosObj</a><br><a href="#">PDNumTreeGetCosObj</a><br><a href="#">PDNumTreesValid</a>                                                                                                                                                                                  |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                      |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | —         | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | —         | M, W    |
| Reader            | —         | —       |

## PDNumTreePut

```
void PDNumTreePut (PDNumTree theTree, ASInt32 key,
 CosObj value);
```

|                        |                                                                                                                                                                                                |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Puts a new entry in the number tree. If an entry with this number is already in the tree, it is replaced.                                                                                      |
| <b>Parameters</b>      | <p><i>theTree</i></p> <p>The number tree for which a new entry is added</p> <p><i>key</i></p> <p>The number of the entry.</p> <p><i>value</i></p> <p>The value associated with <i>key</i>.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                           |
| <b>Exceptions</b>      | None                                                                                                                                                                                           |
| <b>Notifications</b>   | <a href="#">PDNumTreeNumAdded</a>                                                                                                                                                              |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                               |
| <b>Related Methods</b> | <a href="#">PDNumTreeEnum</a><br><a href="#">PDNumTreeGet</a><br><a href="#">PDNumTreeRemove</a>                                                                                               |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                       |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## PDNumTreeRemove

```
void PDNumTreeRemove (PDNumTree theTree, ASInt32 key);
```

|                        |                                                                                                                       |
|------------------------|-----------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Removes the specified object from the tree. Does nothing if no object with that number exists.                        |
| <b>Parameters</b>      | <i>theTree</i><br>The number tree from which an entry is removed.<br><i>key</i><br>The number of the entry to remove. |
| <b>Return Value</b>    | None                                                                                                                  |
| <b>Exceptions</b>      | None                                                                                                                  |
| <b>Notifications</b>   | <a href="#">PDNumTreeNumRemoved</a>                                                                                   |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                      |
| <b>Related Methods</b> | <a href="#">PDNumTreeEnum</a><br><a href="#">PDNumTreeGet</a><br><a href="#">PDNumTreePut</a>                         |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.              |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |



**PDPage**

## PDPAGEAcquirePDEContent

```
PDEContent PDPAGEAcquirePDEContent (PDPAGE page,
ExtensionID clientID);
```

**Description** Creates a *PDEContent* from the *PDPAGE*'s contents and resources. The *PDEContent* is cached, so that subsequent calls on the same *PDPAGE* return the same *PDEContent*, even if the request is from another PDFEdit client. The *PDEContent* remains in the cache as long as someone has it acquired—until someone not using the PDFEdit API changes the *PDPAGE*'s contents, such as the viewer rotating a page 90 degrees.

Do not call [PDERelease](#) on *PDEContent* you have acquired with [PDPAGEAcquirePDEContent](#); call [PDPAGEReleasePDEContent](#) to release it.

**Parameters**

*page*  
The page whose content object is acquired.

*clientID*  
Identifies the caller/client.  
For plug-ins, this should be the [gExtensionID](#) extension.  
For the Adobe PDF Library, if there is only one client of the PDFEdit subsystem, *clientID* should be zero. If there are multiple clients, each should specify a nonzero, non-negative *clientID*. (A negative *clientID* is reserved for the implementation.)

**Return Value** A *PDEContent* representing the page's contents.

**Exceptions** None

**Notifications** None

**Header File** *PagePDECntCalls.h*

**Related Methods** [PDEContentCreateFromCosObj](#)  
[PDPAGEReleasePDEContent](#)  
[PDPAGESetPDEContent](#)

**Availability** Plug-ins: Available if *PI\_PAGE\_PDE\_CONTENT\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## PDPageAddAnnot

```
void PDPageAddAnnot (PDPage aPage, ASInt32 addAfter,
PDAnnot annot);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Adds an annotation at the specified location in a page's annotation array.                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>Parameters</b>      | <p><i>aPage</i></p> <p>The page to which the annotation is added.</p> <p><i>addAfter</i></p> <p>The index into the page's annotation array where the annotation is added. See Section 6.6 in the <a href="#">Portable Document Format Reference Manual</a> for a description of the annotation array. The first annotation in the array has an index of zero. Passing a value of -2 adds the annotation to the end of the array. Passing other negative values produces undefined results.</p> <p><i>annot</i></p> <p>The annotation to add.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>Exceptions</b>      | <p>Raises <a href="#">pdErrOpNotPermitted</a> if:</p> <ol style="list-style-type: none"> <li>1) The annotation is of subtype Text and the document's permissions do not include <i>pdPermEditNotes</i> (see <a href="#">PDPerms</a>).</li> </ol> <p>or</p> <ol style="list-style-type: none"> <li>2) The annotation is of any other subtype and the document's permissions do not include <i>pdPermEdit</i>.</li> </ol>                                                                                                                          |
| <b>Notifications</b>   | <a href="#">PDPageWillAddAnnot</a><br><a href="#">PDPageDidAddAnnot</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Related Methods</b> | <a href="#">PDPageCreateAnnot</a><br><a href="#">PDPageAddNewAnnot</a><br><a href="#">PDPageRemoveAnnot</a>                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>Available in:</b>   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDPPageAddCosContents

```
void PDPPageAddCosContents (PDPPage page, CosObj newContents);
```

|                        |                                                                                                                                                                                                                   |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Completely replaces the contents of the specified page with <i>newContents</i> .                                                                                                                                  |
| <b>Parameters</b>      | <p><i>page</i></p> <p>The page whose Cos contents are replaced.</p> <p><i>newContents</i></p> <p>A stream Cos object or an array Cos object containing the new contents (stream Cos objects) for <i>page</i>.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                              |
| <b>Exceptions</b>      | None                                                                                                                                                                                                              |
| <b>Notifications</b>   | <a href="#">PDDocDidChangePages</a>                                                                                                                                                                               |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                  |
| <b>Related Methods</b> | <a href="#">PDPPageRemoveCosContents</a>                                                                                                                                                                          |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                          |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | —         | —       |

## PDPageAddCosResource

```
void PDPageAddCosResource (PDPage page, char* resType,
 char* resName, CosObj resObj);
```

### Description

Adds a Cos resource to a page object. See Section 7.5 in the [Portable Document Format Reference Manual](#) for a description of page resources.

The necessary dictionaries are created automatically if the page does not already have any resources of the type specified by *resType*, or does not have a Resources dictionary. For example, if you specify a font resource, but the page does not already have a font resource dictionary, this method automatically creates one and puts the font you specify into it.

*ProcSet* resources cannot be added using this method; they must be added using Cos-level methods to:

- 1) Get the page's Resources dictionary
- 2) Get the *ProcSet* array from the Resources dictionary
- 3) Add an element to the *ProcSet* array.

### Parameters

*page*

The page to which a resource is added.

*resType*

Resource type. The named resource types in PDF are: *Font*, *XObject*, *ColorSpace*, *Extended graphics state*, *Pattern*, and *Property list*. Although *ProcSet* is also a resource type, it cannot be added by this method.

*resName*

Resource name. For example, the name of a font might be *F1*.

*resObj*

The Cos object being added as a resource to *page*.

### Return Value

None

### Exceptions

None

### Notifications

[PDDocDidChangePages](#)

### Header File

*PDCalls.h*

**Related Methods**    [PDPageRemoveCosResource](#)  
[PDPageGetCosResources](#)

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |



## PDPageAddNewAnnot

```
PDAnnot PDPageAddNewAnnot (PDPage apage, ASInt32 addAfter,
 ASAtom subType, const ASFixedRect* initialRect);
```

### Description

Adds an annotation to the page. To make the annotation visible after adding it, convert the coordinates of *initialRect* to device coordinates using [AVPageViewRectToDevice](#), then call [AVPageViewInvalidateRect](#) using the converted rectangle.

This method is equivalent to calling [PDPageCreateAnnot](#) followed by [PDPageAddAnnot](#).

### Parameters

*apage*

The page to which the annotation is added.

*addAfter*

Where to add the annotation in the page's annotation array. See Section 6.6 in the [Portable Document Format Reference Manual](#) for a description of the annotation array.

Passing a value of -2 adds the annotation to the end of the array (this is generally what you should do unless you have a need to place the annotation at a special location in the array).

Passing a value of -1 adds the annotation to the beginning of the array.

Passing other negative values produces undefined results.

*subType*

The subtype of the annotation to add.

*initialRect*

Pointer to a rectangle specifying the annotation's bounds, specified in user space coordinates.

### Return Value

The newly-created annotation.

## Exceptions

Raises [pdErrOpNotPermitted](#) if:

- 1) The annotation is of subtype Text and the document's permissions do not include *pdPermEditNotes* (see [PDPerms](#)).
- or
- 2) The annotation is of any other subtype and the document's permissions do not include *pdPermEdit*.

## Notifications

[PDPageWillAddAnnot](#)  
[PDPageDidAddAnnot](#)

## Header File

*PDCalls.h*

## Related Methods

[PDPageCreateAnnot](#)  
[PDPageAddAnnot](#)  
[PDPageRemoveAnnot](#)

## Example

```
AnnotCnt = PDPageGetNumAnnots(TmpPDPage);
myannot= PDPageAddNewAnnot(TmpPDPage,
 AnnotCnt-1,
 ASAtomFromString(STR_EXTENSION_NAME),
 &rect);
/* Display the annotation on the page. */
AVPageViewGetAnnotRect(avPageView,
 ClipAnnotation, &avRect);
AVPageViewInvalidateRect(avPageView,
 &avRect);
```

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDPageCreateAnnot

```
PDAnnot PDPageCreateAnnot (PDPage aPage, ASAtom subtype,
 const ASFixedRect* initialLocation);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                         |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Creates a new annotation, associated with the specified page's <i>CosDoc</i> , but not added to the page. Use <a href="#">PDPageAddAnnot</a> to add the annotation to the page.                                                                                                                                                                                                         |
| <b>Parameters</b>      | <p><i>aPage</i></p> <p>The page to whose <i>CosDoc</i> the annotation is added.</p> <p><i>subtype</i></p> <p>Subtype of annotation to create.</p> <p><i>initialLocation</i></p> <p>Pointer to a rectangle specifying the annotation's bounds, specified in user space coordinates.</p>                                                                                                  |
| <b>Return Value</b>    | The newly-created annotation.                                                                                                                                                                                                                                                                                                                                                           |
| <b>Exceptions</b>      | <p>Raises <a href="#">pdErrOpNotPermitted</a> if:</p> <ol style="list-style-type: none"> <li>1) The annotation is of subtype <i>Text</i> and the document's permissions do not include <i>pdPermEditNotes</i> (see <a href="#">PDPerms</a>).</li> <li>or</li> <li>2) The annotation is of any other subtype and the document's permissions do not include <i>pdPermEdit</i>.</li> </ol> |
| <b>Notifications</b>   | <a href="#">PDAnnotWasCreated</a>                                                                                                                                                                                                                                                                                                                                                       |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                        |
| <b>Related Methods</b> | <a href="#">PDPageAddAnnot</a><br><a href="#">PDPageAddNewAnnot</a>                                                                                                                                                                                                                                                                                                                     |
| <b>Example</b>         | <pre>annot = PDPageCreateAnnot(pdPage,     ASAtomFromString("Text"), &amp;rect); PDPageAddAnnot(annot);</pre>                                                                                                                                                                                                                                                                           |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                                                |
| <b>Available in:</b>   |                                                                                                                                                                                                                                                                                                                                                                                         |

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDPPageDrawContentsToWindow

```
void PDPPageDrawContentsToWindow (PDPPage page, void* window,
 void* displayContext, ASBool isDPS, ASFixedMatrix* matrix,
 ASFixedRect* updateRect, CancelProc cancelProc,
 void* cancelProcClientData);
```

### Description

Draws the contents of a page into the specified window.

This method just draws a bitmap to the window. If you want a live document, you need to open an [AVDoc](#) for the PDF file.

The page can also be derived from a [PDDoc](#).

In UNIX, this method cannot be used to draw into a window. UNIX developers should instead use [AVDocsReadOnly](#) to draw PDF files into their own window from a plug-in.

*Note: This method cannot be reliably used to print to a device.*

### Parameters

*page*

The page to draw into *window*.

*window*

Pointer to a platform-dependent window object (*WindowPtr* or *CWindowPtr* in Mac OS, or *HWND* in Windows).

In Mac OS, to draw into an offscreen GWorld, pass *NULL* in *window* and pass the *GWorldPtr* in *displayContext*.

In Windows, to draw into an offscreen DC, pass *NULL* for *window*.

*displayContext*

A platform-dependent display context structure (*GWorldPtr* in Mac OS, *HDC* in Windows, pointer to an *XBitmapRec* in UNIX). In Mac OS and in UNIX, *displayContext* is ignored if *window* is non-*NULL*.

*Note: displayContext cannot be reliably used as the hDC for a printer device.*

*isDps*

Currently unused. Always set to *false*.

## *matrix*

Pointer to the matrix to concatenate onto the default page matrix. It is useful for converting from page to window coordinates and for scaling.

## *updateRect*

Pointer to the rectangle to draw, defined in user space coordinates. Any objects outside of *updateRect* will not be drawn. All objects are drawn if *updateRect* is *NULL*.

## *cancelProc*

Procedure called periodically to check for user cancel of the drawing operation. The default cancel proc can be obtained using [AVAppGetCancelProc](#). May be *NULL*, in which case no cancel proc is used.

## *cancelProcClientData*

Pointer to user-supplied data to pass to *cancelProc* each time it is called. Should be *NULL* if *cancelProc* is *NULL*.

|                        |                                                                            |
|------------------------|----------------------------------------------------------------------------|
| <b>Return Value</b>    | None                                                                       |
| <b>Exceptions</b>      | None                                                                       |
| <b>Notifications</b>   | None                                                                       |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                           |
| <b>Related Methods</b> | <a href="#">AVAppGetCancelProc</a><br><a href="#">PDDrawCosObjToWindow</a> |

## Example

```

/* Example 1 for UNIX */
XWindowRec winrec;
memset (&winrec, 0, sizeof(winrec));

port = AVWindowGetPlatformThing(win);
winrec.display = XtDisplay(port);
winrec.window = XtWindow(port);
winrec.CMap = None;
winrec.colorStrategy = CS_Find;
PDPPageDrawContentsToWindow(page, winrec,
 displayContext, false, &matrix, 0, 0, 0);

/* Example 2 for Windows */
/* Write the page in the active AVDoc to the
 window */
CancelProc cancelProc;
void* cancelProcData;
HWND window;

AVDoc adoc;
PDDoc pdoc;
PDPPage page;
HDC dc;
ASFixedMatrix matrix;
AVPageView pageview;
ASInt32 pagenum;

cancelProc =
 AVAppGetCancelProc(&cancelProcData);
adoc = AVAppGetActiveDoc();
pdoc = AVDocGetPDDoc(adoc);
pageview = AVDocGetPageView(adoc);
pagenum = AVPageViewGetPageNum(pageview);

page = PDDocAcquirePage(pdoc, pagenum);
AVPageViewGetPageToDevMatrix(pageview,
 &matrix);
dc = GetDC(window);

PDPPageDrawContentsToWindow(page, NULL, (void
 *)dc, false, &matrix, NULL, cancelProc,
 cancelProcData);
PDPPageRelease(page);

```

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDPageEnumContents

```
void PDPageEnumContents (PDPage page, PDGraphicEnumMonitor mon,
 void* clientData);
```

### Description

Enumerates the contents of a page, calling a procedure for each drawing object in the page description.

*Note: This method is provided only for backwards compatibility. It has not been updated beyond PDF Version 1.1 and may not work correctly for newly created PDF 1.2 or later files. You should use the PDFEdit API to enumerate page contents.*

### Parameters

*page*

The page whose contents are enumerated.

*mon*

Pointer to a structure containing user-supplied callbacks that are called for each drawing operator on a page.

Enumeration ends if any procedure returns *false*.

*clientData*

Pointer to user-supplied data to pass to *mon* each time it is called.

### Return Value

None

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

None

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Reader      | M, W, U   | M, W, U |

PDPageEnumResources

```
void PDPageEnumResources (PDPage page,
 PDResourceEnumMonitor mon, void* clientData);
```

**Description** (Obsolete, provided only for backwards compatibility)  
Enumerates the page’s resources, calling an enumeration procedure for each resource.  
Instead of this method, use [PDDocEnumResources](#).  
*Note: This method is provided only for backwards compatibility. It has not been updated beyond PDF Version 1.1 and may not work correctly for newly created PDF 1.2 or later files. You should use the PDFEdit API to enumerate page resources.*

**Parameters**

*page*  
The page whose Cos resources are enumerated.

*mon*  
Pointer to a structure containing user-supplied callbacks that are called for each of the page’s resources.  
Enumeration ends if any procedure returns *false*.

*clientData*  
Pointer to user-supplied data to pass to each procedure in *mon* when it is called.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocEnumResources](#)  
[PDPageGetCosResources](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDPageGetAnnot

[PDAnnot](#) PDPageGetAnnot ([PDPage](#) aPage, ASInt32 annotIndex);

|                        |                                                                                                                                                                                                |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the <i>annotIndex</i> annotation on the page.                                                                                                                                             |
| <b>Parameters</b>      | <i>aPage</i><br>The page on which the annotation is located.<br><i>annotIndex</i><br>The index of the annotation to get on <i>apage</i> . The first annotation on a page has an index of zero. |
| <b>Return Value</b>    | Annotation object.                                                                                                                                                                             |
| <b>Exceptions</b>      | None                                                                                                                                                                                           |
| <b>Notifications</b>   | None                                                                                                                                                                                           |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                               |
| <b>Related Methods</b> | <a href="#">PDPageGetNumAnnots</a>                                                                                                                                                             |

## Example

```

/* select 0th link annot on 0th page
 (assumes that it's a link) */
#define k_Annot
 ASAtomFromString("Annotation")

CosObj tmp;
PDAnnot annot, *annotSel;
PDAction action;

DURING
 PDDocAcquirePage(pdDoc, 0); /* acquire
 0th page */
 annot = PDPAGEGetAnnot(pdPage, 0); /*
 get 0th annot on pdPage */
 AnnotSel = ASmalloc(sizeof(PDAnnot));
 if(annotSel)
 {
 tmp = PDActionGetCosObj(action); /*
 so we can copy its contents */
 annotSel = tmp; / copy the
 contents. The selection server will
 ASfree the pointer when done with
 annotSel */
 AVDocSetSelection(avdoc, k_Annot,
 &annotSel, true);
 }
 PDPAGERelease(pdPage);
HANDLER
 if(pdPage)
 PDPAGERelease(pdPage);
END_HANDLER

```

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDPageGetAnnotIndex

```
ASInt32 PDPageGetAnnotIndex (PDPage aPage, PDAnnot anAnnot);
```

|                        |                                                                                                                                               |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the index of a given annotation object on a given page.                                                                                  |
| <b>Parameters</b>      | <p><i>aPage</i></p> <p>The page to which the annotation is attached.</p> <p><i>anAnnot</i></p> <p>The annotation whose index is obtained.</p> |
| <b>Return Value</b>    | The annotation's index. Returns -1 if the annotation is not on the page.                                                                      |
| <b>Exceptions</b>      | None                                                                                                                                          |
| <b>Notifications</b>   | None                                                                                                                                          |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                              |
| <b>Related Methods</b> | <a href="#">PDPageEnumResources</a><br><a href="#">PDPageGetAnnotSequence</a>                                                                 |
| <b>Example</b>         | <pre>ASInt32 ndx = PDPageGetAnnotIndex(pdPage,     annot);</pre>                                                                              |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                      |

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDPageGetAnnotSequence

```
ASInt32 PDPageGetAnnotSequence (PDPage page, PDAnnot annot);
```

**Description** Returns the index of the specified annotation for the given page. This method is similar to [PDPageGetAnnotIndex](#) but it also checks the annotations information flags to determine whether it is okay provide the index, before actually returning it. The flags are checked to make sure that *PDAnnotOperationSummarize* flag is set, meaning that it is okay to summarize the annotations. The flags are obtained from the annotation handler's [PDAnnotHandlerGetAnnotInfoFlagsProc](#).

**Parameters**

*page*  
The page for which an annotation index is desired.

*annot*  
The annotation for which an index is desired.

**Return Value** The index of the specified annotation or -1 if the annotation is not in the page.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDPageGetAnnotIndex](#)  
[PDPageEnumResources](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

**Available in:**

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | —         | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | —         | M, W    |
| Reader            | —         | —       |

## PDPageGetBBox

```
void PDPageGetBBox (PDPage page, ASFixedRect* bboxP);
```

**Description** Gets the bounding box for a page. The bounding box is the rectangle that encloses all text, graphics, and images on the page.

**Parameters**

*page*

The page whose bounding box is obtained.

*bboxP*

(Filled by the method) Pointer to a rectangle specifying the page's bounding box, specified in user space coordinates.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDPageGetMediaBox](#)  
[PDPageGetCropBox](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDPageGetCosObj

```
CosObj PDPageGetCosObj (PDPage page);
```

|                        |                                                                                                                                                    |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the dictionary Cos object associated with a page. This method does not copy the object, but is instead the logical equivalent of a type cast. |
| <b>Parameters</b>      | <i>page</i><br>The page whose Cos object is obtained.                                                                                              |
| <b>Return Value</b>    | The dictionary Cos object associated with <i>page</i> .                                                                                            |
| <b>Exceptions</b>      | None                                                                                                                                               |
| <b>Notifications</b>   | None                                                                                                                                               |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                   |
| <b>Related Methods</b> | None                                                                                                                                               |
| <b>Availability</b>    | Plug-ins: Available if <i>PL_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                           |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDPageGetCosResources

[CosObj](#) PDPageGetCosResources ( [PDPage](#) page ) ;

**Description** Gets the Cos object corresponding to a page's resource dictionary. A page's resource Cos object may either be directly in the Page Cos object and apply only to the page. Or, it may be in the Pages tree, be shared by multiple pages, and applies to all Page nodes below the point in the Pages tree where it is located.

**Parameters** *page*  
The page whose Cos resources are obtained.

**Return Value** The dictionary Cos object associated with the page's resource.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDPageAddCosResource](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDPageGetCropBox

```
void PDPageGetCropBox (PDPage page, ASFixedRect* cropBoxP);
```

**Description** Gets the crop box for a page. The crop box is the region of the page to display and print.

**Parameters**

*page*

The page whose crop box is being obtained.

*cropBoxP*

(Filled by the method) Pointer to a rectangle specifying the page's crop box, specified in user space coordinates.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods**

[PDPageSetCropBox](#)

[PDPageGetMediaBox](#)

[PDPageGetBBox](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDPageGetDefaultMatrix

```
void PDPageGetDefaultMatrix (PDPage pdpage,
 ASFixedMatrix* defaultMatrix);
```

**Description** Gets the matrix that transforms user space coordinates to rotated and cropped coordinates. The origin of this space is the bottom-left of the rotated, cropped page. Y is increasing.

**Parameters**

*pdpage*  
The page whose default transformation matrix is obtained.

*defaultMatrix*  
(Filled by the method) Pointer to the default transformation matrix.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDPageGetFlippedMatrix](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDPageGetDoc

[PDDoc](#) PDPageGetDoc ( [PDPage](#) page ) ;

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the document that contains the specified page.                                                      |
| <b>Parameters</b>      | <i>page</i><br>The page whose document is obtained.                                                      |
| <b>Return Value</b>    | The document that contains the page.                                                                     |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDFFileSpecGetDoc</a>                                                                        |
| <b>Example</b>         | <pre>PDDoc doc = PDPageGetDoc(pdPage) ;</pre>                                                            |
| <b>Availability</b>    | Plug-ins: Available if <i>PL_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher. |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDPageGetDuration

[ASFixed](#) PDPageGetDuration ( [PDPage](#) page );

|                        |                                                                                                                                                                                                              |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Obtains the page's automatic-advance timing value—the maximum amount of time the page is displayed before the viewer automatically advances to the next page.                                                |
| <b>Parameters</b>      | <p><i>page</i></p> <p>The page whose timing value is obtained.</p>                                                                                                                                           |
| <b>Return Value</b>    | The automatic-advance timing for the page, in seconds. If the page does not have an advance timing value, <i>fxDefaultPageDuration</i> (representing positive infinity, that is, never advance) is returned. |
| <b>Exceptions</b>      | None                                                                                                                                                                                                         |
| <b>Notifications</b>   | None                                                                                                                                                                                                         |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                             |
| <b>Related Methods</b> | <a href="#">PDPageSetDuration</a>                                                                                                                                                                            |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher.                                                                                                     |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |



## PDPageGetFlippedMatrix

```
void PDPageGetFlippedMatrix (PDPage pdpage,
 ASFixedMatrix* flipped);
```

|                        |                                                                                                                                                                                         |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the matrix that transforms user space coordinates to rotated and cropped coordinates. The origin of this space is the top-left of the rotated, cropped page. Y is decreasing.      |
| <b>Parameters</b>      | <p><i>pdpage</i></p> <p>The page whose flipped transformation matrix is obtained.</p> <p><i>flipped</i></p> <p>(Filled by the method) Pointer to the flipped transformation matrix.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                    |
| <b>Exceptions</b>      | None                                                                                                                                                                                    |
| <b>Notifications</b>   | None                                                                                                                                                                                    |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                        |
| <b>Related Methods</b> | <a href="#">PDPageGetDefaultMatrix</a>                                                                                                                                                  |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDPPageGetMediaBox

```
void PDPPageGetMediaBox (PDPPage page, ASFixedRect* mediaBoxP);
```

**Description** Gets the media box for a page. The media box is the “natural size” of the page, for example, the dimensions of an A4 sheet of paper.

**Parameters**

*page*  
The page whose media box is obtained.

*mediaBoxP*  
(Filled by the method) Pointer to a rectangle specifying the page’s media box, specified in user space coordinates.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDPPageSetMediaBox](#)  
[PDPPageGetCropBox](#)  
[PDPPageGetBBox](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDPageGetNumAnnots

```
ASInt32 PDPageGetNumAnnots (PDPage aPage);
```

|                        |                                                                                                                                                                                               |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the number of annotations on a page.                                                                                                                                                     |
| <b>Parameters</b>      | <i>aPage</i><br>The page for which the number of annotations is obtained.                                                                                                                     |
| <b>Return Value</b>    | The number of annotations on <i>aPage</i> .                                                                                                                                                   |
| <b>Exceptions</b>      | None                                                                                                                                                                                          |
| <b>Notifications</b>   | None                                                                                                                                                                                          |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                              |
| <b>Related Methods</b> | <a href="#">PDPageEnumResources</a>                                                                                                                                                           |
| <b>Example</b>         | <pre>AnnotCnt = PDPageGetNumAnnots( TmpPDPage );<br/>myannot= PDPageAddNewAnnot( TmpPDPage ,<br/>    AnnotCnt-1 ,<br/>    ASAtomFromString( STR_EXTENSION_NAME ) ,<br/>    &amp;rect );</pre> |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                      |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDPageGetNumber

```
ASInt32 PDPageGetNumber (PDPage page);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the page number for the specified page.                                                             |
| <b>Parameters</b>      | <i>page</i><br>The page whose page number is obtained.                                                   |
| <b>Return Value</b>    | The page within the document. The first page is 0.                                                       |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDPageNumFromCosObj</a>                                                                      |
| <b>Example</b>         | <pre>if ( !PDPageGetNumber (pdPage) )<br/>    AVAlertNote ( "First page!" );</pre>                       |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher. |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDPageGetPDEContentFilters

```
ASBool PDPageGetPDEContentFilters (PDPage page,
 ASInt32* numFilters, ASAtom** filters);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets filters used by <a href="#">PDPageSetPDEContent</a> .<br>The caller is responsible for allocating the filter array <i>filters</i> that receives the filters. <i>filters</i> can be <i>NULL</i> to just obtain the number of filters.                                                                                                                      |
| <b>Parameters</b>      | <p><i>page</i><br/>The page whose content filters are obtained.</p> <p><i>numFilters</i><br/>(Filled by the method) Number of filters used by <a href="#">PDPageSetPDEContent</a>.</p> <p><i>filters</i><br/>(Filled by the method) Filters used by <a href="#">PDPageSetPDEContent</a>. If <i>NULL</i>, <i>numFilters</i> contains the number of filters.</p> |
| <b>Return Value</b>    | <i>true</i> if filters are obtained, <i>false</i> if page's contents are not cached.                                                                                                                                                                                                                                                                           |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                           |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                           |
| <b>Header File</b>     | <i>PagePDECntCalls.h</i>                                                                                                                                                                                                                                                                                                                                       |
| <b>Related Methods</b> | <a href="#">PDPageSetPDEContent</a><br><a href="#">PDPageSetPDEContentFilters</a>                                                                                                                                                                                                                                                                              |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PAGE_PDE_CONTENT_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                                                                              |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | —         | M, W    |
| Reader            | —         | —       |



## PDPPageGetPDEContentFlags

```
ASBool PDPPageGetPDEContentFlags (PDPPage page, ASUns32* flags);
```

|                        |                                                                                                                                                                       |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets flags used by <a href="#">PDPPageSetPDEContent</a> .                                                                                                             |
| <b>Parameters</b>      | <p><i>page</i></p> <p>The page whose content flags are obtained.</p> <p><i>flags</i></p> <p>(Filled by the method) <a href="#">PDEContentToCosObjFlags</a> flags.</p> |
| <b>Return Value</b>    | <i>true</i> if flags obtained, <i>false</i> if page's contents are not cached.                                                                                        |
| <b>Exceptions</b>      | None                                                                                                                                                                  |
| <b>Notifications</b>   | None                                                                                                                                                                  |
| <b>Header File</b>     | <i>PagePDECntCalls.h</i>                                                                                                                                              |
| <b>Related Methods</b> | <a href="#">PDPPageSetPDEContent</a><br><a href="#">PDPPageSetPDEContentFlags</a>                                                                                     |
| <b>Availability</b>    | Plug-ins: Available if<br><i>PI_PAGE_PDE_CONTENT_VERSION</i> (in <i>PIRequir.h</i> ) is<br>set to 0x00040000 or higher.                                               |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## PDPageGetRotate

[PDRotate](#) PDPageGetRotate ( [PDPage](#) page ) ;

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the rotation value for a page.                                                                      |
| <b>Parameters</b>      | <i>page</i><br>The page whose rotation is obtained.                                                      |
| <b>Return Value</b>    | Rotation value for the given page. Must be one of the <a href="#">PDRotate</a> values.                   |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDPageSetRotate</a>                                                                          |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher. |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDPageGetTransition

[PDTrans](#) PDPageGetTransition ([PDPage](#) page);

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the transition for a given page.                                                                    |
| <b>Parameters</b>      | <i>page</i><br>The page whose transition is obtained.                                                    |
| <b>Return Value</b>    | The page's transition. If the page has no transition, returns a <i>NULL</i> transition.                  |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDPageSetTransition</a><br><a href="#">PDTransNull</a>                                       |
| <b>Availability</b>    | Plug-ins: Available if <i>PL_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher. |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDPageHasTransition

```
ASBool PDPageHasTransition (PDPage page);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Tests whether a page has a transition.                                                                   |
| <b>Parameters</b>      | <i>page</i><br>The page to test.                                                                         |
| <b>Return Value</b>    | <i>true</i> if the page has a transition, <i>false</i> otherwise.                                        |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | None                                                                                                     |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher. |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDPPageNotifyContentsDidChange

```
void PDPPageNotifyContentsDidChange (PDPPage page);
```

**Description**

Broadcasts a [PDPPageContentsDidChange](#) notification. If the Acrobat viewer is version 2.1 or later, also broadcasts a [PDPPageContentsDidChangeEx](#) notification with *invalidateViews* set to *true*.

You must use this method after using Cos methods to change a page's contents. Do not use this method if you use [PDPPageAddCosContents](#) or [PDPPageRemoveCosContents](#) to change a page's contents, because those methods automatically generate the appropriate notifications.

Use [PDPPageNotifyContentsDidChangeEx](#) instead of this method if you wish to suppress the Acrobat viewer's immediate redraw of the page.

**Parameters**

*page*

The page that changed.

**Return Value**

None

**Exceptions**

None

**Notifications**

[PDPPageContentsDidChange](#)  
[PDPPageContentsDidChangeEx](#) (in version 2.1 and later)

**Header File**

*PDCalls.h*

**Related Methods**

[PDPPageAddCosContents](#)  
[PDPPageRemoveCosContents](#)  
[PDPPageNotifyContentsDidChangeEx](#)

**Availability**

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |



## PDPPageNotifyContentsDidChangeEx

```
void PDPPageNotifyContentsDidChangeEx (PDPPage page,
 ASBool invalidateViews);
```

### Description

Broadcasts a [PDPPageContentsDidChange](#) notification and a [PDPPageContentsDidChangeEx](#) notification. These notify the Acrobat viewer that a page's contents have been modified, and tells the Acrobat viewer whether or not to redraw the page immediately.

You must use this method after using Cos methods to change a page's contents. Do not use this method if you use [PDPPageAddCosContents](#) or [PDPPageRemoveCosContents](#) to change a page's contents, because those methods automatically generate the appropriate notifications.

If your plug-in must be compatible with version 2.0 of the Acrobat viewer, you must use [PDPPageNotifyContentsDidChange](#) instead.

### Parameters

*page*

The page that changed.

*invalidateViews*

*true* if the Acrobat viewer redraws the page view, *false* otherwise. This allows plug-ins to make a sequence of modifications to a page's contents, without having the entire page flash after each modification.

Passing *true* for *invalidateViews* is equivalent to calling [PDPPageNotifyContentsDidChange](#).

### Return Value

None

### Exceptions

None

### Notifications

[PDPPageContentsDidChange](#)  
[PDPPageContentsDidChangeEx](#)

### Header File

*PDCalls.h*

### Related Methods

[PDPPageAddCosContents](#)  
[PDPPageRemoveCosContents](#)  
[PDPPageNotifyContentsDidChange](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020001 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDPageNumFromCosObj

```
ASInt32 PDPageNumFromCosObj (CosObj pageObj);
```

|                        |                                                                                                                                            |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the page number for the specified page.                                                                                               |
| <b>Parameters</b>      | <i>pageObj</i><br>Dictionary Cos object for the page whose number is obtained.                                                             |
| <b>Return Value</b>    | The page within the document (the first page in a document is page number zero). Returns -1 if <i>pageObj</i> is a <i>NULL</i> Cos object. |
| <b>Exceptions</b>      | None                                                                                                                                       |
| <b>Notifications</b>   | None                                                                                                                                       |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                           |
| <b>Related Methods</b> | <a href="#">PDPageGetNumber</a>                                                                                                            |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                   |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDPPagePDEContentWasChanged

```
void PDPPagePDEContentWasChanged (PDPPage page,
 ExtensionID clientID);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Indicates a page's <a href="#">PDEContent</a> has changed.<br>Call this after you alter a <i>PDPPage</i> 's <a href="#">PDEContent</a> but do not call <a href="#">PDPPageSetPDEContent</a> , so others who have acquired the <a href="#">PDEContent</a> know it has changed.                                                                                                                                                                                            |
| <b>Parameters</b>      | <i>page</i><br>The page whose content was changed.<br><br><i>clientID</i><br>Identifies the caller/client.<br>For plug-ins, this should be the <a href="#">gExtensionID</a> extension.<br>For the Adobe PDF Library, if there is only one client of the PDFEdit subsystem, <i>clientID</i> should be zero. If there are multiple clients, each should specify a nonzero, non-negative <i>clientID</i> . (A negative <i>clientID</i> is reserved for the implementation.) |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Notifications</b>   | <a href="#">PagePDEContentDidChange</a>                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Header File</b>     | <i>PagePDECntCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Related Methods</b> | <a href="#">PDPPageSetPDEContent</a>                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PAGE_PDE_CONTENT_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                                                                                                                                                                                        |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |



|             |           |     |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader      | —         | —   |

## PDPageRegisterForPDEContentChanged

```
void PDPageRegisterForPDEContentChanged
 (PagePDEContentDidChangeNPROTO proc, ExtensionID clientID);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Registers for the <a href="#">PagePDEContentDidChange</a> notification.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Parameters</b>      | <p><i>proc</i></p> <p>Callback for function to call when an acquired <i>PDPage</i>'s <a href="#">PDEContent</a> has changed.</p> <p><i>clientID</i></p> <p>Identifies the caller/client.</p> <p>For plug-ins, this should be the <a href="#">gExtensionID</a> extension.</p> <p>For the Adobe PDF Library, if there is only one client of the PDFEdit subsystem, <i>clientID</i> should be zero. If there are multiple clients, each should specify a nonzero, non-negative <i>clientID</i>. (A negative <i>clientID</i> is reserved for the implementation.)</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Notifications</b>   | <a href="#">PagePDEContentDidChange</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Header File</b>     | <i>PagePDECntCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Related Methods</b> | <a href="#">PDPageRegisterForPDEContentNotCached</a><br><a href="#">PDPageUnRegisterForPDEContentChanged</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PAGE_PDE_CONTENT_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |



## PDPageRegisterForPDEContentNotCached

```
void PDPageRegisterForPDEContentNotCached
 (PagePDEContentNotCachedNPROTO proc, ExtensionID clientID);
```

### Description

Register for the [PagePDEContentNotCached](#) notification.

This notification is also sent when others change (or delete) a *PDPage*'s contents without using PDFEdit methods. For instance, rotating or deleting a page in the viewer results in this notification being sent.

PDFEdit registers for almost a half dozen different notifications for the different ways Exchange can alter page contents—you may need only this notification.

### Parameters

*proc*

Callback for function to call when an acquired *PDPage*'s [PDEContent](#) is no longer valid.

*clientID*

Identifies the caller/client.

For plug-ins, this should be the [gExtensionID](#) extension.

For the Adobe PDF Library, if there is only one client of the PDFEdit subsystem, *clientID* should be zero. If there are multiple clients, each should specify a nonzero, non-negative *clientID*. (A negative *clientID* is reserved for the implementation.)

### Return Value

None

### Exceptions

None

### Notifications

[PagePDEContentNotCached](#)

### Header File

*PagePDECntCalls.h*

### Related Methods

[PDPageRegisterForPDEContentChanged](#)  
[PDPageUnRegisterForPDEContentNotCached](#)

### Availability

Plug-ins: Available if *PI\_PAGE\_PDE\_CONTENT\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## PDPageRelease

```
void PDPageRelease (PDPage page);
```

|                        |                                                                       |
|------------------------|-----------------------------------------------------------------------|
| <b>Description</b>     | Decrements the specified page's reference count.                      |
| <b>Parameters</b>      | <i>page</i><br>The page whose reference count is decremented.         |
| <b>Return Value</b>    | None                                                                  |
| <b>Exceptions</b>      | None                                                                  |
| <b>Notifications</b>   | None                                                                  |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                      |
| <b>Related Methods</b> | <a href="#">PDBeadAcquirePage</a><br><a href="#">PDDocAcquirePage</a> |

**Example**

```
dest = PDDocAcquirePage(pdDoc, ival-1);
newview = PDViewDestCreate(pdDoc, dest,
 k_XYZ, &destRect, fixedZero, ival-1);
newAction = PDActionNewFromDest(pdDoc,
 newview, pdDoc);
PDPageRelease(dest);
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDPageReleasePDEContent

```
ASInt32 PDPageReleasePDEContent (PDPage page,
 ExtensionID clientID);
```

**Description**

Decrements a *PDPage*'s [PDEContent](#) internal reference count.

The *PDEContent* is *not* automatically deleted when the reference count becomes zero—it remains in the cache until the cache slot is needed for another *PDPage*.

Thus, you don't need to keep a *PDEContent* acquired for performance reasons. There is a notification you can register for, that is sent when a *PDEContent* is actually removed from the cache, thus enabling the use of PDFEdit's tagging methods [PDEAddTag](#), [PDEGetTag](#), [PDERemoveTag](#) on the *PDEContent* object.

**Parameters**

*page*

The page whose content object's use count is decremented.

*clientID*

Identifies the caller/client.

For plug-ins, this should be the [gExtensionID](#) extension.

For the Adobe PDF Library, if there is only one client of the PDFEdit subsystem, *clientID* should be zero. If there are multiple clients, each should specify a nonzero, non-negative *clientID*. (A negative *clientID* is reserved for the implementation.)

**Return Value**

Updated reference count.

**Exceptions**

None

**Notifications**

None

**Header File**

*PagePDECntCalls.h*

**Related Methods**

[PDPageAcquirePDEContent](#)  
[PDPageSetPDEContent](#)

**Availability**

Plug-ins: Available if *PI\_PAGE\_PDE\_CONTENT\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |



## PDPageRemoveAnnot

```
void PDPageRemoveAnnot (PDPage aPage, ASInt32 annotIndex);
```

### Description

Removes an annotation from the specified page. Annotations are stored in Cos arrays, which are automatically compressed when an annotation is removed (see [CosArrayRemove](#)). For this reason, if you use a loop in which you remove annotations, structure the code so the loop processes from the highest to the lowest index. If you loop the other direction, you will skip over annotations immediately following ones you remove.

### Parameters

*aPage*

The page from which the annotation is removed.

*annotIndex*

The index (into the page's annotation array) of the annotation to remove.

### Return Value

None

### Exceptions

Raises [pdErrOpNotPermitted](#) if:

1) The annotation is of subtype *Text* and the document's permissions do not include *pdPermEditNotes* (see [PDPerms](#)).

or

2) The annotation is of any other subtype and the document's permissions do not include *pdPermEdit*.

### Notifications

[PDPageWillRemoveAnnot](#)  
[PDPageDidRemoveAnnot](#)

### Header File

*PDCalls.h*

### Related Methods

[PDPageGetAnnotIndex](#)  
[PDPageAddNewAnnot](#)  
[PDPageAddAnnot](#)

## Example

```
for (i=PDPageGetNumAnnots(pdPage)-1; i>=0; i--)
{
 annot = PDPageGetAnnot(pdPage, i);
 if (PDAnnotIsValid(annot) &&
 PDAnnotGetSubtype(annot) ==
 ASAtomFromString("Link"))
 {
 PDPageRemoveAnnot(annot);
 }
}
```

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0  |
|-------------|-----------|------|
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDPPageRemoveCosContents

```
void PDPPageRemoveCosContents (PDPPage page);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Removes the contents of the specified page.                                                              |
| <b>Parameters</b>      | <i>page</i><br>The page whose Cos contents are removed.                                                  |
| <b>Return Value</b>    | None                                                                                                     |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | <a href="#">PDDocDidChangePages</a>                                                                      |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDPPageAddCosContents</a>                                                                    |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher. |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDPageRemoveCosResource

```
void PDPageRemoveCosResource (PDPage page, char* resType,
 char* resName);
```

**Description** Removes a Cos resource from a page object. See Section 7.5 in the [Portable Document Format Reference Manual](#) for a description of page resources.

**Parameters**

*page*  
The page whose Cos resources are removed.

*resType*  
Resource type. The named resource types in PDF are: *ProcSet*, *Font*, *XObject*, *ColorSpace*, *Extended graphics state*, *Pattern*, and *Property list*.

*resName*  
Resource name. For example, the name of a font might be *F1*.

**Return Value** None

**Exceptions** None

**Notifications** [PDDocDidChangePages](#)

**Header File** *PDCalls.h*

**Related Methods** [PDPageAddCosResource](#)  
[PDPageGetCosResources](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | —         | —       |

## PDPageSetCropBox

```
void PDPageSetCropBox (PDPage page, ASFixedRect cropBox);
```

**Description** Sets the crop box for a page. The crop box is the region of the page to display and print. This method ignores the request if either the width or height of cropBox is less than 72 points (one inch).

**Parameters**

*page*  
The page whose crop box is being set.

*cropBox*  
Rectangle specifying the page's crop box, specified in user space coordinates.

**Return Value** None

**Exceptions** None

**Notifications** [PDDocWillChangePages](#)  
[PDDocDidChangePages](#)

**Header File** *PDCalls.h*

**Related Methods** [PDPageGetCropBox](#)  
[PDPageSetMediaBox](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDPageSetDuration

```
void PDPageSetDuration (PDPage page, ASFixed fxDuration);
```

**Description** Sets the page's automatic-advance timing value—the maximum amount of time the page is displayed before the viewer automatically advances to the next page.

**Parameters**

*page*  
The page whose timing is set.

*fxDuration*  
The auto-advance timing, in seconds. If no advance timing is desired, *fxDuration* should be set to *fxDefaultPageDuration*.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDPageGetDuration](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

**Available in:**

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | —         | —       |

## PDPageSetMediaBox

```
void PDPageSetMediaBox (PDPage page, ASFixedRect mediaBox);
```

**Description** Sets the media box for a page. The media box is the “natural size” of the page, for example, the dimensions of an A4 sheet of paper.

**Parameters**

*page*  
The page whose media box is set.

*mediaBox*  
Rectangle specifying the page’s media box, specified in user space coordinates.

**Return Value** None

**Exceptions** None

**Notifications** [PDDocWillChangePages](#)  
[PDDocDidChangePages](#)

**Header File** *PDCalls.h*

**Related Methods** [PDPageGetMediaBox](#)  
[PDPageSetCropBox](#)  
[PDPageGetBBox](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0  |
|-------------|-----------|------|
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDPageSetPDEContent

```
ASBool PDPageSetPDEContent (PDPage page,
 ExtensionID clientID);
```

**Description** Sets the page's [PDEContent](#) back into the *PDPage*'s Cos object, using the same compression filters with which the content was previously encoded.

This method calls [PDPageNotifyContentsDidChangeEx](#).

**Parameters**

*page*

The page whose [PDEContent](#) is set.

*clientID*

Identifies the caller/client.

For plug-ins, this should be the [gExtensionID](#) extension.

For the Adobe PDF Library, if there is only one client of the PDFEdit subsystem, *clientID* should be zero. If there are multiple clients, each should specify a nonzero, non-negative *clientID*. (A negative *clientID* is reserved for the implementation.)

**Return Value** *true* if [PDEContent](#) successfully set, *false* otherwise.

**Exceptions** None

**Notifications** [PDPageContentsDidChangeEx](#)

**Header File** *PagePDECntCalls.h*

**Related Methods** [PDEContentToCosObj](#)  
[PDPageAcquirePDEContent](#)  
[PDPageGetPDEContentFlags](#)

**Availability** Plug-ins: Available if *PI\_PAGE\_PDE\_CONTENT\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |



| Plug-in API | 3.0, 3.01 | 4.0  |
|-------------|-----------|------|
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## PDPageSetPDEContentFilters

```
ASBool PDPageSetPDEContentFilters (PDPage page,
 ASInt32 numFilters, ASAtom* filters);
```

|                        |                                                                                                                                                                                                                                      |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Sets filters used by <a href="#">PDPageSetPDEContent</a> . The filters are not instantiated until <a href="#">PDPageSetPDEContent</a> is called.                                                                                     |
| <b>Parameters</b>      | <i>page</i><br>The page whose content filters are set.<br><i>numFilters</i><br>Number of filters used by <a href="#">PDPageSetPDEContent</a> .<br><i>filters</i><br>Array of filters to use by <a href="#">PDPageSetPDEContent</a> . |
| <b>Return Value</b>    | <i>true</i> if filters were set, <i>false</i> if page's contents are not cached (and so nothing was done).                                                                                                                           |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                 |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                 |
| <b>Header File</b>     | <i>PagePDECntCalls.h</i>                                                                                                                                                                                                             |
| <b>Related Methods</b> | <a href="#">PDPageGetPDEContentFilters</a><br><a href="#">PDPageSetPDEContent</a>                                                                                                                                                    |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PAGE_PDE_CONTENT_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                    |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## PDPageSetPDEContentFlags

```
ASBool PDPageSetPDEContentFlags (PDPage page, ASUns32 flags);
```

**Description** Sets flags used by [PDPageSetPDEContent](#). The flags are not instantiated until [PDPageSetPDEContent](#) is called.

**Parameters** *page*  
The page whose content flags are set.

*flags*  
[PDEContentToCosObjFlags](#) flags. The following flags are ignored, since content is always a page:

*kPDEContentToForm*

*kPDEContentToCharProc*

**Return Value** *true* if flags were set, *false* if page's contents are not cached (and so nothing was done).

**Exceptions** None

**Notifications** None

**Header File** *PagePDECntCalls.h*

**Related Methods** [PDPageGetPDEContentFlags](#)  
[PDPageSetPDEContent](#)

**Availability** Plug-ins: Available if *PI\_PAGE\_PDE\_CONTENT\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

## PDPageSetRotate

```
void PDPageSetRotate (PDPPage page, PDRotate angle);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Sets the rotation value for a page.                                                                      |
| <b>Parameters</b>      | <p><i>page</i></p> <p>The page whose rotation is set.</p>                                                |
| <b>Return Value</b>    | None                                                                                                     |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | <a href="#">PDDocWillChangePages</a><br><a href="#">PDDocDidChangePages</a>                              |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDPageGetRotate</a>                                                                          |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher. |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | —         | —       |

## PDPageSetTransition

```
void PDPageSetTransition (PDPage page, PDTrans pdt);
```

**Description** Sets the transition for a given page.

**Parameters**

*page*  
The page whose transition is set.

*pdt*  
The transition for the page.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDPageGetTransition](#)  
[PDTransNull](#)

**Example**

```
/* To remove the transition from a page,
 pass a NULL transition: */
```

```
PDPageSetTransition(page, PDTransNull());
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDPageStmGetInlineImage

```
ASUns32 PDPageStmGetInlineImage (ASStm stm, ASUns32 flags,
 CosDoc cosDoc, CosObj resDict, PDPageStmImageDataProc proc,
 void* procClientData, ASUns32* imageRawDataStmOffsetP,
 ASUns32* imageRawDataLenP, CosObj* imageDict);
```

### Description

Reads a PDF page content inline image from a stream. The stream is typically obtained by getting the Cos stream for a page contents or a Form contents, and calling [CosStreamOpenStm](#) to open the stream using the “filtered” mode. This method is called after a **BI** token has been read from the stream. **BI** indicates that the following tokens comprise an inline image dictionary and data.

Begins reading at the current *stm* position. Returns the number of bytes read. This is the number of bytes read from the stream and indicates the amount by which the stream position has advanced.

The image attributes dictionary is returned in *imageDict*. The image data is passed to the [PDPageStmImageDataProc](#); if *proc* is not provided, the image data is discarded.

*imageRawDataStmOffsetP* and *imageRawDataLenP* may be *NULL*, in which case they are ignored.

The caller should call [CosObjDestroy](#) on *imageDict* when done.

### Parameters

*stm*

The stream from which data is read.

*cosDoc*

The *CosDoc* with the *PDPage* that contains the inline image.

*resDict*

The Resources dict in which to look up ColorSpace resources for inline images.

*flags*

Currently unused by this method. (Used by [PDPageStmGetToken](#).)

*proc*

Callback method to handle inline image data.

*procClientData*

Client data passed to *proc*.

*imageRawDataStmOffsetP*

(Filled by the method) Offset of the data stream, after **BI**, relative to the beginning of *stm*.

*imageRawDataLenP*

(Filled by the method) Offset of the last byte of the data stream between the **BI** and **EI** PDF operators.

*imageDict*

(Filled by the method) The returned image dictionary.

**Return Value** Number of bytes read from *stm*.

**Exceptions** Can raise memory, I/O, and parsing exceptions.

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [CosStreamOpenStm](#)  
[PDPageGetCosObj](#)  
[PDPageStmGetToken](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDPPageStmGetToken

```
ASUns32 PDPPageStmGetToken (ASStm stm, ASUns32 flags,
PDPPageStmStringOverflowProc proc, void* procClientData,
PDPPageStmToken pageStmToken);
```

### Description

Reads a PDF page content token from a stream. The stream is typically obtained by getting the Cos stream for a page contents or a Form contents, and calling [CosStreamOpenStm](#) to open the stream using the “filtered” mode.

It begins reading at the current stream position, and reads exactly one token. It returns the number of bytes read. This is the number of bytes read from the stream and indicates the amount by which the stream position has advanced.

If the token is an integer, real (*ASFixed*) or *ASBool*, then the value is returned in *pageStmToken.iVal*. If the token is a string or a name, the value is returned in *pageStmToken.sVal*, and the length of the token is in *pageStmToken.sValLen*. Strings are not *NULL*-terminated, but names are *NULL*-terminated. If a string length is greater than *kPDPPageStmStringMax*, the [PDPPageStmStringOverflowProc](#) is called repeatedly with portions of the string. On return from [PDPPageStmGetToken](#), the value of *pageStmToken.sValLen* is zero, and *pageStmToken.sVal* is empty (*iVal*, *sVal* and *sValLen* are components of the [PDPPageStmToken](#)). If there is no overflow proc, then the first *kPDPPageStmStringMax* bytes of the string will be returned in *pageStmToken.sVal*, and the remaining bytes are lost. The value of *pageStmToken.sValLen* is *kPDPPageStmStringMax* in this case.

If the token is **BI** (begin inline image), [PDPPageStmGetInlinelImage](#) should be called to parse the inline image.

### Parameters

*stm*

The stm from which data is read.



## *flags*

Bit field of options such as “skip comments” token the returned token value (set the size field before calling).

*kPDPPageStmSkipComments* — Skip comments during tokenizing.

## *proc*

Callback method to handle long strings.

## *procClientData*

Client data passed to the callback method.

## *pageStmToken*

The returned token.

## **Return Value**

Number of bytes read from *stm*.

## **Exceptions**

Can raise memory, I/O, and parsing exceptions.

## **Notifications**

None

## **Header File**

*PDCalls.h*

## **Related Methods**

[CosStreamOpenStm](#)  
[PDPageGetCosObj](#)  
[PDPageStmGetInlinImage](#)

## **Availability**

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

## **Available in:**

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDPageUnRegisterForPDEContentChanged

```
void PDPageUnRegisterForPDEContentChanged
 (PagePDEContentDidChangeNPROTO proc, ExtensionID clientID);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Unregisters for the <a href="#">PagePDEContentDidChange</a> notification.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>Parameters</b>      | <p><i>proc</i></p> <p>Callback for function to call when an acquired <i>PDPage</i>'s <a href="#">PDEContent</a> has changed.</p> <p><i>clientID</i></p> <p>Identifies the caller/client.</p> <p>For plug-ins, this should be the <a href="#">gExtensionID</a> extension.</p> <p>For the Adobe PDF Library, if there is only one client of the PDFEdit subsystem, <i>clientID</i> should be zero. If there are multiple clients, each should specify a nonzero, non-negative <i>clientID</i>. (A negative <i>clientID</i> is reserved for the implementation.)</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Notifications</b>   | <a href="#">PagePDEContentDidChange</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Header File</b>     | <i>PagePDECntCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Related Methods</b> | <a href="#">PDPageRegisterForPDEContentChanged</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PAGE_PDE_CONTENT_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |



## PDPageUnRegisterForPDEContentNotCached

```
void PDPageUnRegisterForPDEContentNotCached
 (PagePDEContentNotCachedNPROTO proc, ExtensionID clientID);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Unregisters for the <a href="#">PagePDEContentNotCached</a> notification.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Parameters</b>      | <p><i>proc</i></p> <p>Callback for function to call when an acquired <i>PDPage</i>'s <a href="#">PDEContent</a> is no longer valid.</p> <p><i>clientID</i></p> <p>Identifies the caller/client.</p> <p>For plug-ins, this should be the <a href="#">gExtensionID</a> extension.</p> <p>For the Adobe PDF Library, if there is only one client of the PDFEdit subsystem, <i>clientID</i> should be zero. If there are multiple clients, each should specify a nonzero, non-negative <i>clientID</i>. (A negative <i>clientID</i> is reserved for the implementation.)</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Notifications</b>   | <a href="#">PagePDEContentNotCached</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Header File</b>     | <i>PagePDECntCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Related Methods</b> | <a href="#">PDPageRegisterForPDEContentNotCached</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PAGE_PDE_CONTENT_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |



**PDPageLabel**

## PDPageLabelEqual

```
ASBool PDPageLabelEqual (PDPageLabel pdlOne,
 PDPageLabel pdlTwo);
```

|                        |                                                                                                                                                                                    |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Compares two page labels to see if they are equivalent. Two labels are equivalent if they have the same style, starting page, and prefix strings which are the same byte-for-byte. |
| <b>Parameters</b>      | <i>pdlOne</i><br>A page label.<br><i>pdlTwo</i><br>Another page label.                                                                                                             |
| <b>Return Value</b>    | <i>true</i> if the two labels are valid and equivalent, <i>false</i> otherwise.                                                                                                    |
| <b>Exceptions</b>      | None                                                                                                                                                                               |
| <b>Notifications</b>   | None                                                                                                                                                                               |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                   |
| <b>Related Methods</b> | <a href="#">PDPageLabellsValid</a><br><a href="#">PDPageLabelNew</a>                                                                                                               |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                           |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDPageLabelFromCosObj

[PDPageLabel](#) PDPageLabelFromCosObj ([CosObj](#) cosLabel);

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Creates a type cast of the <i>cosObj</i> to a <i>PDPageLabel</i> object.                                 |
| <b>Parameters</b>      | <i>cosLabel</i><br>Cos object level representation of a page label.                                      |
| <b>Return Value</b>    | Page label representation of <i>cosLabel</i> .                                                           |
| <b>Exceptions</b>      | Raises <a href="#">pdErrBadBaseObj</a> if <i>cosLabel</i> is not a valid page label.                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDPageLabelGetCosObj</a>                                                                     |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher. |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |



## PDPageLabelGetCosObj

[CosObj](#) PDPageLabelGetCosObj ( [PDPageLabel](#) pdl ) ;

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Creates a type cast of the page label object to a Cos object.                                            |
| <b>Parameters</b>      | <i>pdل</i><br>A <i>PDPageLabel</i> representation of a page label.                                       |
| <b>Return Value</b>    | A <i>CosObj</i> representation of <i>pdل</i> if the page label is valid, <i>NULL CosObj</i> otherwise.   |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDPageLabelFromCosObj</a>                                                                    |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher. |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDPageLabelGetPrefix

```
const char* PDPageLabelGetPrefix (PDPageLabel pgLabel,
 ASInt32* prefixLen);
```

**Description** Returns the prefix string for the label. The prefix string is transitory and should be copied immediately.

**Parameters** *pgLabel*

Label for page whose prefix is desired.

*prefixLen*

The length, in bytes, of the prefix string. Zero if page label is not valid.

**Return Value** The prefix string for the label, or *NULL* if none is specified.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDPageLabelGetStart](#)  
[PDPageLabelGetStyle](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

**Available in:**

| Library API       | 1.0 | 4.0     |
|-------------------|-----|---------|
| Adobe PDF Library | —   | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDPPageLabelGetStart

```
ASInt32 PDPPageLabelGetStart (PDPPageLabel pgLabel);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the starting page number for the page label specified.                                              |
| <b>Parameters</b>      | <p><i>pgLabel</i></p> <p>Page label for page whose start page number is desired.</p>                     |
| <b>Return Value</b>    | Returns the starting number for the label. Returns 1 if the page label is not valid, or unknown.         |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDPPageLabelGetPrefix</a><br><a href="#">PDPPageLabelGetStyle</a>                            |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher. |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | —         | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | —         | M, W    |
| Reader            | —         | M, W, U |

## PDPageLabelGetStyle

[ASAtom](#) PDPageLabelGetStyle ( [PDPageLabel](#) pgLabel );

|                        |                                                                                                           |
|------------------------|-----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Returns an <i>ASAtom</i> for the style of the label.                                                      |
| <b>Parameters</b>      | <i>pgLabel</i><br>Page label whose style is desired.                                                      |
| <b>Return Value</b>    | An <i>ASAtom</i> for the label style. If no style is specified, returns <i>ASAtomFromString("None")</i> . |
| <b>Exceptions</b>      | Raises an exception if storage is exhausted or file access fails.                                         |
| <b>Notifications</b>   | None                                                                                                      |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                          |
| <b>Related Methods</b> | <a href="#">PDPageLabelGetPrefix</a><br><a href="#">PDPageLabelGetStart</a>                               |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.  |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDPPageLabellsValid

```
ASBool PDPPageLabelIsValid (PDPPageLabel pgLabel);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Determines whether a page label is valid.                                                                |
| <b>Parameters</b>      | <i>pgLabel</i><br>Page label whose validity is determined.                                               |
| <b>Return Value</b>    | <i>true</i> if the label is valid, <i>false</i> otherwise.                                               |
| <b>Exceptions</b>      | Raises an exception if storage is exhausted or file access fails.                                        |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDPPageLabelEqual</a>                                                                        |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher. |

### Available in:

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | —         | M, W    |
| Reader      | —         | M, W, U |

## PDPageLabelNew

```
PDPageLabel PDPageLabelNew (PDDoc pdDoc, ASAtom style,
const char* prefix, ASInt32 prefixLen, ASInt32 startAt);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Constructs a new label object in the document with the specified style, prefix, and starting page number.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Parameters</b>      | <p><i>pdDoc</i></p> <p>The document that contains the new page label.</p> <p><i>style</i></p> <p>(<i>Optional</i>) Specifies the numbering system to use for the numeric portion of each label in this range of pages. Possible values are <b>D</b> for decimal numbers, <b>R</b> for upper-case Roman numbers, <b>r</b> for lower-case Roman numbers, <b>A</b> for upper-case alphabetic numbers, or <b>a</b> for lower-case alphabetic numbers. If this key is not present, the labels for this range will not have a numeric portion.</p> <p><i>prefix</i></p> <p>(<i>Optional</i>) Specifies a string to prefix to the numeric portion of the page label.</p> <p><i>prefixLen</i></p> <p>Length in bytes of the prefix string.</p> <p><i>startAt</i></p> <p>(<i>Optional</i>) Specifies the value to use when generating the numeric portion of the first label in this range. This number must be greater than or equal to 1. The default value is 1.</p> |
| <b>Return Value</b>    | The newly-created page label.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Exceptions</b>      | Raises <a href="#">pdErrBadBaseObj</a> if the base pages object is missing or invalid.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>Related Methods</b> | <a href="#">PDDocRemovePageLabel</a><br><a href="#">PDDocGetPageLabel</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00040000 or higher.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

**Available in:**

|                   |     |         |
|-------------------|-----|---------|
| Library API       | 1.0 | 4.0     |
| Adobe PDF Library | —   | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | —         | M, W |
| Reader      | —         | —    |

**PDPath**



## PDPathEnum

```
void PDPathEnum (PDPath obj, PDPathEnumMonitor mon,
void* clientData);
```

**Description** Enumerates the specified path's operators, calling *one of several* user-supplied callbacks for each operator. The callback that is called depends on which operator is encountered.

**Parameters**

*obj*  
The path whose operators are enumerated.

*mon*  
Pointer to a structure that contains callbacks. One of the callbacks will be called for each path segment operator in the path.  
Enumeration ends if any of the monitor's callbacks returns *false*.

*clientData*  
Pointer to user-supplied data to pass to the monitor's callbacks each time one is called.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** None

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |



## PDPathGetPaintOp

```
ASInt32 PDPathGetPaintOp (PDPath obj);
```

|                        |                                                                                                                                                                                                                                   |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets flags that indicate which paint/close/clip operators are used for the specified path. For a description of the path painting operators, see Section 8.6.2 in the <a href="#">Portable Document Format Reference Manual</a> . |
| <b>Parameters</b>      | <p><i>obj</i></p> <p>The path whose painting operators are obtained.</p>                                                                                                                                                          |
| <b>Return Value</b>    | A bitwise OR of the <a href="#">PDPathPaintOp</a> flags.                                                                                                                                                                          |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                              |
| <b>Notifications</b>   | None                                                                                                                                                                                                                              |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                  |
| <b>Related Methods</b> | None                                                                                                                                                                                                                              |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                          |

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

**PDStyle**

## PDStyleGetColor

```
void PDStyleGetColor (PDStyle obj, PDColorValue color);
```

|                        |                                                                                                                                                                       |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets a style's color.                                                                                                                                                 |
| <b>Parameters</b>      | <p><i>obj</i></p> <p>The style whose color is obtained.</p> <p><i>color</i></p> <p>(Filled by the method) Pointer to a structure that contains the style's color.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                  |
| <b>Exceptions</b>      | None                                                                                                                                                                  |
| <b>Notifications</b>   | None                                                                                                                                                                  |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                      |
| <b>Related Methods</b> | <a href="#">PDWordGetNthCharStyle</a>                                                                                                                                 |
| <b>Example</b>         | <pre>PDColorValueRec theColor;  PDStyle style = PDWordGetNthCharStyle(pdWF,     pdWord, 0); PDStyleGetColor(style, &amp;theColor);</pre>                              |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                              |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDStyleGetFont

[PDFont](#) PDStyleGetFont ([PDStyle](#) obj);

|                        |                                                                                                              |
|------------------------|--------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the specified style's font.                                                                             |
| <b>Parameters</b>      | <i>obj</i><br>The style whose font is obtained.                                                              |
| <b>Return Value</b>    | The font for the specified style.                                                                            |
| <b>Exceptions</b>      | None                                                                                                         |
| <b>Notifications</b>   | None                                                                                                         |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                             |
| <b>Related Methods</b> | <a href="#">PDWordGetNthCharStyle</a>                                                                        |
| <b>Example</b>         | <pre>PDStyle style = PDWordGetNthCharStyle(pdWF,<br/>    pdWord, 0);<br/>font = PDStyleGetFont(style);</pre> |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.     |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDStyleGetFontSize

[ASFixed](#) PDStyleGetFontSize ([PDStyle](#) obj);

**Description** Get a style's font size.

**Parameters** *obj*  
The style whose font size is obtained.

**Return Value** The size of the font, in points.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDWordGetNthCharStyle](#)

**Example**

```
PDStyle style = PDWordGetNthCharStyle(pdWF,
 pdWord, 0);
size = PDStyleGetFontSize(style);
PDFontGetName(font, mbuf, sizeof(mbuf));
if(size == fixedTen && !strstr(mbuf,
 "Italic"))
{
 /* process the 10 pt. Italic */
 ...
}
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

**PDText**



## PDTextEnum

```
void PDTextEnum (PDText text, PDStringEnumProc enumProc,
void* clientData);
```

**Description** Enumerates the strings of a text object, calling a procedure for each string. The [PDText](#) object may be obtained from the [PDGraphicEnumTextProc](#) callback of [PDGraphicEnumMonitor](#).

**Parameters**

*text*

The text object whose strings are enumerated.

*enumProc*

User-supplied callback to call for each text string in the text object.

Enumeration ends if *enumProc* returns *false*.

*clientData*

Pointer to user-supplied data to pass to *enumProc* each time it is called.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** None

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDTextGetState

```
void PDTextGetState (PDText obj, PDTextStateP stateP,
 ASInt32 stateLen);
```

|                        |                                                                                                                                                                                                                                                                                       |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the text state for a text object. See Sections 8.2 and 8.7 in the <a href="#">Portable Document Format Reference Manual</a> for a discussion of the text state parameters.                                                                                                       |
| <b>Parameters</b>      | <p><i>obj</i></p> <p>The text object whose text state is obtained.</p> <p><i>stateP</i></p> <p>(Filled by the method) Pointer to a <a href="#">PDTextState</a> structure containing the text state information.</p> <p><i>stateLen</i></p> <p>Must be <i>sizeof(PDTextState)</i>.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                  |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                  |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                  |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                      |
| <b>Related Methods</b> | None                                                                                                                                                                                                                                                                                  |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                              |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

**PDTextAnnot**

## PDTextAnnotGetContents

```
ASInt32 PDTextAnnotGetContents (PDTextAnnot aTextAnnot,
 char* buffer, ASInt32 bufSize);
```

### Description

Gets the text of a text annotation.

*Note: For Roman viewers, this text is always stored in the PDFDocEncoding. For non-Roman character set viewers, this text is stored as PDFDocEncoding or Unicode, depending on the file's creator. Files created in a non-Roman environment contain Unicode versions of these strings; in a Roman environment, files contain PDFDocEncoding versions of these strings.*

### Parameters

*aTextAnnot*

The text annotation whose text is obtained.

*buffer*

*(Filled by the method)* A buffer into which the text is placed. The text is encoded using PDFDocEncoding, and can be converted to a platform's native encoding using [PDXlateToHost](#) or [PDXlateToHostEx](#).

*bufSize*

The maximum number of characters that *buffer* can hold.

### Return Value

The number of characters written into *buffer*.

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDTextAnnotSetContents](#)  
[PDXlateToPDFDocEnc](#)  
[PDXlateToPDFDocEncEx](#)  
[PDXlateToHost](#)  
[PDXlateToHostEx](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDTextAnnotIsOpen

```
ASBool PDTextAnnotIsOpen (PDTextAnnot aTextAnnot);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Tests whether or not a text annotation is open.                                                          |
| <b>Parameters</b>      | <i>aTextAnnot</i><br>The text annotation whose open/closed state is obtained.                            |
| <b>Return Value</b>    | <i>true</i> if the annotation is open, <i>false</i> otherwise.                                           |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDTextAnnotSetOpen</a>                                                                       |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher. |

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDTextAnnotSetContents

```
void PDTextAnnotSetContents (PDTextAnnot aTextAnnot,
 const char* str, ASInt32 nBytes);
```

### Description

Sets the text of a text annotation.

*Note: For Roman viewers, this text is always stored in the PDFDocEncoding. For non-Roman character set viewers, this text is stored as PDFDocEncoding or Unicode, depending on the file's creator. Files created in a non-Roman environment contain Unicode versions of these strings; in a Roman environment, files contain PDFDocEncoding versions of these strings.*

### Parameters

*aTextAnnot*

The text annotation whose text is set.

*str*

A string containing the new text. The string must be encoded using PDFDocEncoding. Strings in a platform's native encoding can be converted to PDFDocEncoding using [PDXlateToPDFDocEnc](#) or [PDXlateToPDFDocEncEx](#).

*nBytes*

The number of characters in *str*.

### Return Value

None

### Exceptions

None

### Notifications

[PDAnnotWillChange](#)  
[PDAnnotDidChange](#)

### Header File

*PDCalls.h*

### Related Methods

[PDTextAnnotGetContents](#)  
[PDXlateToPDFDocEnc](#)  
[PDXlateToPDFDocEncEx](#)  
[PDXlateToHost](#)  
[PDXlateToHostEx](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |



## PDTextAnnotSetOpen

```
void PDTextAnnotSetOpen (PDTextAnnot aTextAnnot, ASBool isOpen);
```

|                        |                                                                                                                                                              |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Opens or closes a text annotation.                                                                                                                           |
| <b>Parameters</b>      | <i>aTextAnnot</i><br>The annotation to open or close.<br><i>isOpen</i><br><i>true</i> if the annotation is opened, <i>false</i> if the annotation is closed. |
| <b>Return Value</b>    | None                                                                                                                                                         |
| <b>Exceptions</b>      | None                                                                                                                                                         |
| <b>Notifications</b>   | <a href="#">PDAnnotWillChange</a><br><a href="#">PDAnnotDidChange</a>                                                                                        |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                             |
| <b>Related Methods</b> | <a href="#">PDTextAnnotIsOpen</a>                                                                                                                            |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                     |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## **PDTextSelect**

## PDTextSelectCreatePageHilite

```
PDTextSelect PDTextSelectCreatePageHilite (PDPage page,
HiliteEntry* list, ASInt32 listLen);
```

### Description

Creates a text selection from a page and a list of highlights specified as character offsets from the start of the page. Character offsets are a well-defined quantity in the PDF file, and are therefore stable against revisions of the word-finding algorithm, which makes them a good way to isolate yourself from changes in the algorithm.

This method does not highlight the text selection. That occurs when you pass the *PDTextSelect* returned by this method to [AVDocSetSelection](#).

*Note: As in the Acrobat viewer, the text selection is always of whole words, not part of words.*

### Parameters

*page*

The page on which the highlights appear.

*list*

Pointer to an array of highlight entries. If the length field of a *HiliteEntry* is 0, the entire word is highlighted. *list* should not contain multiple instances of the same highlight; the display appearance is undefined when it does.

*listLen*

The number of highlight entries in *list*.

### Return Value

The newly-created text selection.

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDTextSelectCreateWordHilite](#)  
[PDTextSelectCreateRanges](#)  
[PDDocCreateStructTreeRoot](#)  
[PDWordGetCharOffset](#)  
[AVDocSetSelection](#)  
[PDTextSelectDestroy](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDTextSelectCreateRanges

```
PDTextSelect PDTextSelectCreateRanges (PDPage page,
PDTextSelectRange range, ASInt32 rangeCount);
```

**Description** Creates a text selection from one or more ranges.  
This method does not highlight the text selection. That occurs when you pass the *PDTextSelect* returned by this method to [AVDocSetSelection](#).

**Parameters**

*page*  
The page on which the text appears.

*range*  
Pointer to an array of ranges that are used to create a text selection. Each array element is a [PDTextSelectRange](#) structure.

*rangeCount*  
The number of ranges in *range*.

**Return Value** A text selection created from the specified ranges.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [AVDocSetSelection](#)  
[PDTextSelectCreatePageHilite](#)  
[PDTextSelectCreateWordHilite](#)  
[PDTextSelectDestroy](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDTextSelectCreateWordHilite

```
PDTextSelect PDTextSelectCreateWordHilite (PDPage page,
HiliteEntry* list, ASInt32 listLen);
```

### Description

Creates a text selection from a list of highlights specified as word offsets from the start of the page. Word offsets are not well-defined in PDF files, but are calculated by the word-finding algorithm. As a result, word offsets will in general differ in different versions of the word finding algorithm. If you choose to store word offsets, you must also store the version of the word-finding algorithm from which they are obtained using [PDWordFinderGetLatestAlgVersion](#).

This method does not highlight the text selection. That occurs when you pass the *PDTextSelect* returned by this method to [AVDocSetSelection](#).

### Parameters

*page*

The page on which the highlights appear.

*list*

Pointer to an array of highlight entries. *list* should not contain multiple instances of the same highlight; the display appearance is undefined when *it does*.

*listLen*

The number of highlight entries in *list*.

### Return Value

The newly-created text selection.

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDTextSelectCreateRanges](#)  
[PDTextSelectCreatePageHilite](#)  
[PDWordFinderGetNthWord](#)  
[PDWordFinderGetLatestAlgVersion](#)  
[AVDocSetSelection](#)  
[PDTextSelectDestroy](#)

### Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDTextSelectDestroy

```
void PDTextSelectDestroy (PDTextSelect text);
```

**Description** Deletes a text selection object (the text on the page remains unchanged). Do not use this method to destroy a text selection that was passed to [AVDocSetSelection](#); such text selections are automatically destroyed when a new selection is made or the selection is cleared.

**Parameters** *text*  
The text selection to destroy.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocCreateStructTreeRoot](#)  
[PDTextSelectCreatePageHilite](#)  
[PDTextSelectCreateRanges](#)  
[PDTextSelectCreateWordHilite](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDTextSelectEnumQuads

```
void PDTextSelectEnumQuads (PDTextSelect text,
 PDTextSelectEnumQuadProc proc, void* procObj);
```

**Description** Enumerates the bounding quads in a text selection. *proc* is called for each quad. If a word is on a curve it may have a quad for each character, but it may also have two characters per quad. An upright word will have only one quad for all the characters. An upright hyphenated word will have two quads.

**Parameters**

*text*  
The text selection whose bounding quads are enumerated.

*proc*  
User-supplied callback to call for each quad. Enumeration halts if *proc* returns *false*.

*procObj*  
User-supplied data to pass to *proc* each time it is called.

**Return Value** None

**Exceptions** [genErrBadParm](#)

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocCreateStructTreeRoot](#)  
[PDTextSelectCreatePageHilite](#)  
[PDTextSelectCreateRanges](#)  
[PDTextSelectCreateWordHilite](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Reader      | M, W, U   | M, W, U |

## PDTextSelectEnumText

```
void PDTextSelectEnumText (PDTextSelect text,
PDTextSelectEnumTextProc proc, void* procObj);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Enumerates the strings of the specified text select object, calling a procedure for each string. A string, in this context, is the set of like-styled characters within a word. It is never larger than a single word. A word containing three styles is enumerated as three strings. There is no guaranteed correspondence between these strings and the actual show strings in the PDF file. The version 2.0 Acrobat viewers enumerate text in the order it appears in the PDF file, which is often not the same as the order in which a person would read the text. |
| <b>Parameters</b>      | <p><i>text</i></p> <p>The text selection whose strings are enumerated.</p> <p><i>proc</i></p> <p>User-supplied callback to call for each string in the text object.</p> <p>Enumeration ends if <i>proc</i> returns <i>false</i>.</p> <p><i>procObj</i></p> <p>User-supplied data to pass to <i>proc</i> each time it is called.</p>                                                                                                                                                                                                                                    |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| <b>Exceptions</b>      | <a href="#">genErrBadParm</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>Related Methods</b> | <a href="#">PDDocCreateStructTreeRoot</a><br><a href="#">PDTextSelectCreatePageHilite</a><br><a href="#">PDTextSelectCreateRanges</a><br><a href="#">PDTextSelectCreateWordHilite</a>                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>Available in:</b>   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDTextSelectGetBoundingRect

```
void PDTextSelectGetBoundingRect (PDTextSelect text,
 ASFixedRect* boundRectP);
```

**Description** Gets a text selection's bounding rectangle. This is the smallest rectangle that completely encloses all characters in the selection.

**Parameters**

*text*

The text selection whose bounding rectangle is determined.

*boundRectP*

(Filled by the method) Pointer to the text selection's bounding rectangle, specified in user space coordinates.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocCreateStructTreeRoot](#)  
[PDTextSelectCreatePageHilite](#)  
[PDTextSelectCreateRanges](#)  
[PDTextSelectCreateWordHilite](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDTextSelectGetPage

```
ASInt32 PDTextSelectGetPage (PDTextSelect text);
```

|                        |                                                                                                                                                                                       |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the page number of a text selection's page.                                                                                                                                      |
| <b>Parameters</b>      | <i>text</i><br>The text selection whose page number is obtained.                                                                                                                      |
| <b>Return Value</b>    | The page number of the text selection's page.                                                                                                                                         |
| <b>Exceptions</b>      | None                                                                                                                                                                                  |
| <b>Notifications</b>   | None                                                                                                                                                                                  |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                      |
| <b>Related Methods</b> | <a href="#">PDDocCreateStructTreeRoot</a><br><a href="#">PDTextSelectCreatePageHilite</a><br><a href="#">PDTextSelectCreateRanges</a><br><a href="#">PDTextSelectCreateWordHilite</a> |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                              |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDTextSelectGetRange

```
void PDTextSelectGetRange (PDTextSelect textP, ASInt32 index,
 PDTextSelectRange range);
```

|                        |                                                                                                                                                                                                                                                  |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Extracts the range specified by <i>index</i> from a text selection. Use <a href="#">PDTextSelectGetRangeCount</a> to determine the number of ranges in a text selection.                                                                         |
| <b>Parameters</b>      | <i>textP</i><br>The text selection from which a range is extracted.<br><i>index</i><br>The index of the range to extract from <i>textP</i> .<br><i>range</i><br>(Filled by the method) Pointer to a structure that contains the specified range. |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                             |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                             |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                             |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                 |
| <b>Related Methods</b> | <a href="#">PDTextSelectGetRangeCount</a><br><a href="#">PDDocCreateStructTreeRoot</a><br><a href="#">PDTextSelectCreatePageHilite</a><br><a href="#">PDTextSelectCreateRanges</a><br><a href="#">PDTextSelectCreateWordHilite</a>               |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                         |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDTextSelectGetRangeCount

```
ASInt32 PDTextSelectGetRangeCount (PDTextSelect textP);
```

|                        |                                                                                                                                                                                                                               |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the number of ranges in a text selection. Use <a href="#">PDTextSelectGetRange</a> to extract a single range from a text selection.                                                                                      |
| <b>Parameters</b>      | <i>textP</i><br>The text selection whose range count is obtained.                                                                                                                                                             |
| <b>Return Value</b>    | The number of ranges in the text selection.                                                                                                                                                                                   |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                          |
| <b>Notifications</b>   | None                                                                                                                                                                                                                          |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                              |
| <b>Related Methods</b> | <a href="#">PDTextSelectGetRange</a><br><a href="#">PDDocCreateStructTreeRoot</a><br><a href="#">PDTextSelectCreatePageHilite</a><br><a href="#">PDTextSelectCreateRanges</a><br><a href="#">PDTextSelectCreateWordHilite</a> |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                      |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## **PDThread**

## PDThreadDestroy

```
void PDThreadDestroy (PDThread thread);
```

**Description** Deletes an article thread from its document. You must call [PDDocRemovePageLabel](#) to remove the thread from the document (if it was added to one using [PDDocAddThread](#)) before calling *PDThreadDestroy*.

**Parameters** *thread*  
The thread to destroy.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDThreadNew](#)  
[PDThreadFromCosObj](#)  
[PDDocRemovePageLabel](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDThreadFromCosObj

```
PDThread PDThreadFromCosObj (CosObj obj);
```

|                        |                                                                                                                                                               |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the thread object corresponding to the specified Cos object. This method does not copy the object, but is instead the logical equivalent of a type cast. |
| <b>Parameters</b>      | <i>obj</i><br>Dictionary Cos object for the thread.                                                                                                           |
| <b>Return Value</b>    | The <i>PDThread</i> object for the thread.                                                                                                                    |
| <b>Exceptions</b>      | Raises <a href="#">pdErrBadThread</a> if the thread is not valid as defined by <a href="#">PDThreadIsValid</a> .                                              |
| <b>Notifications</b>   | None                                                                                                                                                          |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                              |
| <b>Related Methods</b> | <a href="#">PDThreadGetCosObj</a>                                                                                                                             |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                      |

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDThreadGetCosObj

[CosObj](#) PDThreadGetCosObj ([PDThread](#) thread);

|                        |                                                                                                                                                                                                                              |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the Cos object corresponding to a thread. This method does not copy the object, but is instead the logical equivalent of a type cast.                                                                                   |
| <b>Parameters</b>      | <p><i>thread</i></p> <p>The thread whose Cos object is to be obtained.</p>                                                                                                                                                   |
| <b>Return Value</b>    | Dictionary Cos object for the thread. The contents of the dictionary can be enumerated using <a href="#">CosObjEnum</a> . Returns a <i>NULL</i> Cos object if <a href="#">PDThreadIsValid(thread)</a> returns <i>false</i> . |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                         |
| <b>Notifications</b>   | None                                                                                                                                                                                                                         |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                             |
| <b>Related Methods</b> | <a href="#">CosObjEnum</a><br><a href="#">PDThreadFromCosObj</a>                                                                                                                                                             |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                     |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDThreadGetFirstBead

```
PDBead PDThreadGetFirstBead (PDThread thread);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets an article thread's first bead.                                                                     |
| <b>Parameters</b>      | <p><i>thread</i></p> <p>The thread whose first bead is obtained.</p>                                     |
| <b>Return Value</b>    | The thread's first bead, or a <i>NULL</i> Cos object if the thread has no beads.                         |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | <a href="#">PDThreadSetFirstBead</a><br><a href="#">PDBeadGetNext</a><br><a href="#">PDBeadGetPrev</a>   |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher. |

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDThreadGetInfo

```
ASInt32 PDThreadGetInfo (PDThread thread, char* infoKey,
 char* buffer, ASInt32 bufSize);
```

**Description**

Gets the specified article thread's info.

*Note: For Roman viewers, this text is always stored in the PDFDocEncoding. For non-Roman character set viewers, this text is stored as PDFDocEncoding or Unicode, depending on the file's creator. Files created in a non-Roman environment contain Unicode versions of these strings; in a Roman environment, files contain PDFDocEncoding versions of these strings.*

**Parameters**

*thread*

The thread whose thread info is obtained.

*infoKey*

The key whose value is obtained.

*buffer*

*(Filled by the method)* The value associated with *infoKey*.

*bufSize*

The maximum number of characters that *buffer* can hold.

**Return Value**

The number of characters written into *buffer*.

**Exceptions**

None

**Notifications**

None

**Header File**

*PDCalls.h*

**Related Methods**

[PDThreadSetInfo](#)

## Example

```
size = PDDocGetNumThreads(pdDoc);
if(size)
{
 for(i = 0;i<size;i++)
 {
 /* get the right thread */
 thread = PDDocGetThread(pdDoc, i);
 len = PDThreadGetInfo(thread, "Title",
 msg, sizeof(msg));
 if(!strcmp(msg, "Contents"))
 break;
 }
}

size = PDDocGetNumThreads(pdDoc);
if(size)
{
 for(i = 0;i<size;i++)
 {
 /* get the right thread */
 thread = PDDocGetThread(pdDoc, i);
 len = PDThreadGetInfo(thread, "Title",
 msg, sizeof(msg));
 if(!strcmp(msg, "Contents"))
 PDThreadSetInfo(thread, "Title",
 "In Progress", sizeof(
 "In Progress"));
 }
}
```

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDThreadIsValid

```
ASBool PDThreadIsValid (PDThread thread);
```

**Description** Tests whether or not a thread is valid. This is intended only to ensure that the thread has not been deleted, not to ensure that all necessary information is present and valid.

**Parameters** *thread*  
The thread whose validity is tested.

**Return Value** *true* if the thread is valid, *false* otherwise.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** None

**Example**

```
size = PDDocGetNumThreads(pdDoc);
if(size){
 for(i = 0;i<size;i++){/* get the right
 thread */
 thread = PDDocGetThread(pdDoc, i);
 if(!PDThreadIsValid(thread))
 AVAlertNote("invalid thread");
 }
}
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDThreadNew

```
PDThread PDThreadNew (PDDoc aDocP);
```

**Description** Creates a new article thread. Use [PDDocAddThread](#) to add the thread to a document.

**Parameters** *doc*  
The document in which the thread is created.

**Return Value** The newly-created thread.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDThreadDestroy](#)  
[PDThreadFromCosObj](#)  
[PDDocAddThread](#)

**Example**

```
PDThread thread = PDThreadNew(doc);
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | —         | —       |

## PDThreadSetFirstBead

```
void PDThreadSetFirstBead (PDThread thread,
 PDBead newFirstBead);
```

|                        |                                                                                                                                |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Sets an article thread's first bead.                                                                                           |
| <b>Parameters</b>      | <i>thread</i><br>The thread whose first bead is set.<br><i>newFirstBead</i><br>The bead to use as the first in <i>thread</i> . |
| <b>Return Value</b>    | None                                                                                                                           |
| <b>Exceptions</b>      | None                                                                                                                           |
| <b>Notifications</b>   | None                                                                                                                           |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                               |
| <b>Related Methods</b> | <a href="#">PDThreadGetFirstBead</a><br><a href="#">PDBeadInsert</a>                                                           |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                       |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDThreadSetInfo

```
void PDThreadSetInfo (PDThread thread, char* infoKey,
 char* buffer, ASInt32 bufSize);
```

**Description**

Sets the specified article thread's info.

*Note: For Roman viewers, this text is always stored in the PDFDocEncoding. For non-Roman character set viewers, this text is stored as PDFDocEncoding or Unicode, depending on the file's creator. Files created in a non-Roman environment contain Unicode versions of these strings; in a Roman environment, files contain PDFDocEncoding versions of these strings.*

**Parameters**

*thread*

The thread whose thread info is set.

*infoKey*

The key whose value is set.

*buffer*

The value to set.

*bufSize*

The number of characters in *buffer*.

**Return Value**

None

**Exceptions**

None

**Notifications**

[PDThreadDidChange](#)

**Header File**

*PDCalls.h*

**Related Methods**

[PDThreadGetInfo](#)

## Example

```
size = PDDocGetNumThreads(pdDoc);
if(size)
{
 for(i = 0;i<size;i++)
 {
 /* get the right thread */
 thread = PDDocGetThread(pdDoc, i);
 len = PDThreadGetInfo(thread, "Title",
 msg, sizeof(msg));
 if(!strcmp(msg, "Contents"))
 break;
 }
}

size = PDDocGetNumThreads(pdDoc);
if(size)
{
 for(i = 0;i<size;i++)
 {
 /* get the right thread */
 thread = PDDocGetThread(pdDoc, i);
 len = PDThreadGetInfo(thread, "Title",
 msg, sizeof(msg));
 if(!strcmp(msg, "Contents"))
 PDThreadSetInfo(thread, "Title",
 "In Progress", sizeof(
 "In Progress"));
 }
}
```

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0  |
|-------------|-----------|------|
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## **PDTrans**

## PDTransEqual

```
ASBool PDTransEqual (PDTrans pdtOne, PDTrans pdtTwo);
```

**Description** Tests two transitions for equality. Two transitions are equal if and only if their Cos objects are equal (see [CosObjEqual](#)).

**Parameters** *pdtOne, pdtTwo*  
The transitions to test for equality.

**Return Value** *true* if the transitions are equal, *false* otherwise.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [CosObjEqual](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDTransFromCosObj

```
PDTrans PDTransFromCosObj (CosObj obj);
```

|                        |                                                                                                                                                                                                    |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Converts the specified dictionary Cos object to a transition and verifies that the transition is valid. This method does not copy the object but is instead the logical equivalent of a type cast. |
| <b>Parameters</b>      | <i>obj</i><br>Dictionary Cos object to convert to a transition.                                                                                                                                    |
| <b>Return Value</b>    | The transition corresponding to the given dictionary object, <i>obj</i> .                                                                                                                          |
| <b>Exceptions</b>      | Raises <a href="#">pdErrBadBaseObj</a> if the transition is not valid as determined by <a href="#">PDTransIsValid</a> .                                                                            |
| <b>Notifications</b>   | None                                                                                                                                                                                               |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                   |
| <b>Related Methods</b> | <a href="#">PDTransGetCosObj</a><br><a href="#">PDTransIsValid</a>                                                                                                                                 |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher.                                                                                           |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |



## PDTransGetCosObj

```
CosObj PDTransGetCosObj (PDTrans pdt);
```

|                        |                                                                                                                                                                                              |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the dictionary Cos object corresponding to the transition and verifies that the transition is valid. This method does not copy the object but is the logical equivalent of a type cast. |
| <b>Parameters</b>      | <i>pdt</i><br>The transition whose Cos object is obtained.                                                                                                                                   |
| <b>Return Value</b>    | The dictionary Cos object corresponding to the transition. Returns the <i>NULL</i> Cos object if the transition is invalid as determined by <a href="#">PDTransIsValid</a> .                 |
| <b>Exceptions</b>      | None                                                                                                                                                                                         |
| <b>Notifications</b>   | None                                                                                                                                                                                         |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                             |
| <b>Related Methods</b> | <a href="#">PDTransFromCosObj</a><br><a href="#">PDTransIsValid</a>                                                                                                                          |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher.                                                                                     |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDTransGetDuration

[ASFixed](#) PDTransGetDuration ([PDTrans](#) pdt);

|                        |                                                                                                                                                                                                                                                                                                             |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Obtains the duration for the given transition.                                                                                                                                                                                                                                                              |
| <b>Parameters</b>      | <p><i>pdt</i></p> <p>The transition for which the duration is obtained.</p>                                                                                                                                                                                                                                 |
| <b>Return Value</b>    | <p>The transition duration, specified in seconds. If no duration is specified in the transition, the return value is <i>fxDefaultTransDuration</i> (1 second).</p> <p><i>Note: Standard durations are</i></p> <p><i>0 seconds - fast</i></p> <p><i>1 second - medium</i></p> <p><i>2 seconds - slow</i></p> |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                        |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                        |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                            |
| <b>Related Methods</b> | None                                                                                                                                                                                                                                                                                                        |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher.                                                                                                                                                                                                    |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDTransGetSubtype

[ASAtom](#) PDTransGetSubtype ( [PDTrans](#) pdt );

|                        |                                                                                                                     |
|------------------------|---------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets a transition's subtype.                                                                                        |
| <b>Parameters</b>      | <i>pdt</i><br>The transition whose subtype is obtained.                                                             |
| <b>Return Value</b>    | The <i>ASAtom</i> for the transition's subtype. Can be converted to a string using <a href="#">ASAtomGetCount</a> . |
| <b>Exceptions</b>      | None                                                                                                                |
| <b>Notifications</b>   | None                                                                                                                |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                    |
| <b>Related Methods</b> | <a href="#">ASAtomGetCount</a>                                                                                      |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher.            |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDTransIsValid

```
ASBool PDTransIsValid (PDTrans pdt);
```

|                        |                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Tests whether a transition is valid, that is, the transition has not been deleted.                       |
| <b>Parameters</b>      | <i>pdt</i><br>The transition dictionary whose validity is tested.                                        |
| <b>Return Value</b>    | <i>true</i> if the transition is valid, <i>false</i> otherwise.                                          |
| <b>Exceptions</b>      | None                                                                                                     |
| <b>Notifications</b>   | None                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                         |
| <b>Related Methods</b> | None                                                                                                     |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher. |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDTransNew

```
PDTrans PDTransNew (PDDoc pdd, ASAtom asaSubtype,
ASFixed fxDuration);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Creates a new transition of the specified type and duration associated with the <i>CosDoc</i> of the given <i>PDDoc</i> .                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>Parameters</b>      | <p><i>pdd</i></p> <p>The <i>PDDoc</i> to whose <i>CosDoc</i> the transition is added.</p> <p><i>asaSubtype</i></p> <p>The transition subtype to create. Must be one of the transition effects described in Section 6.4 in the <a href="#">Portable Document Format Reference Manual</a>. All implementations that support transitions are required to support these transitions at a minimum. Plug-ins can register new types or provide a new handler for an existing type.</p> <p><i>fxDuration</i></p> <p>The transition duration, in seconds.</p> |
| <b>Return Value</b>    | The newly-created transition.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>Exceptions</b>      | Raises <a href="#">pdErrBadBaseObj</a> if the transition is not valid as determined by <a href="#">PDTransIsValid</a> .                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Related Methods</b> | <a href="#">PDTransIsValid</a><br><a href="#">PDTransNewFromCosDoc</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher.                                                                                                                                                                                                                                                                                                                                                                                                                                              |

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0  |
|-------------|-----------|------|
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDTransNewFromCosDoc

```
PDTrans PDTransNewFromCosDoc (CosDoc cd, ASAtom asaSubtype,
 ASFixed fxDuration);
```

|                        |                                                                                                                                                                                                                                                                                                                                          |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Creates a new transition of the specified type and duration associated with the given <i>CosDoc</i> .                                                                                                                                                                                                                                    |
| <b>Parameters</b>      | <p><i>cd</i></p> <p>The <i>CosDoc</i> to which the transition is added.</p> <p><i>asaSubtype</i></p> <p>The transition subtype to create. This subtype is returned by the <a href="#">AVTransHandlerGetTypeProc</a> callback in <a href="#">AVTransHandler</a>.</p> <p><i>fxDuration</i></p> <p>The transition duration, in seconds.</p> |
| <b>Return Value</b>    | The newly-created transition.                                                                                                                                                                                                                                                                                                            |
| <b>Exceptions</b>      | Raises <a href="#">pdErrBadBaseObj</a> if the transition is not valid as determined by <a href="#">PDTransIsValid</a> .                                                                                                                                                                                                                  |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                         |
| <b>Related Methods</b> | <a href="#">PDTransIsValid</a><br><a href="#">PDTransNew</a>                                                                                                                                                                                                                                                                             |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher.                                                                                                                                                                                                                                 |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDTransNull

[PDTrans](#) PDTransNull (void);

|                        |                                                                                                                                                        |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Obtains a <i>NULL</i> transition. This can be used in conjunction with <a href="#">PDTransEqual</a> to determine whether a transition is <i>NULL</i> . |
| <b>Parameters</b>      | None                                                                                                                                                   |
| <b>Return Value</b>    | A <i>NULL</i> transition.                                                                                                                              |
| <b>Exceptions</b>      | None                                                                                                                                                   |
| <b>Notifications</b>   | None                                                                                                                                                   |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                       |
| <b>Related Methods</b> | <a href="#">PDTransEqual</a>                                                                                                                           |
| <b>Example</b>         | <pre>PDTrans trans = PDPageGetTransition(page); if (!PDTransEqual(PDTransNull(), trans)) {     /* Page has a transition */ }</pre>                     |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020002 or higher.                                               |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |



## **PDViewDest**

## PDViewDestCreate

```
PDViewDestination PDViewDestCreate (PDDoc doc,
 PDPage initialPage, ASAtom initialFitType,
 const ASFixedRectP initialRect, const ASFixed initialZoom,
 ASInt32 pageNumber);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Creates a new view destination object.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>Parameters</b>      | <p><i>doc</i></p> <p>The document in which the destination is used.</p> <p><i>initialPage</i></p> <p>The destination page.</p> <p><i>initialFitType</i></p> <p>Destination fit type. Must be one of the <a href="#">View Destination Fit Types</a>, which must be converted into an <i>ASAtom</i> using <a href="#">ASAtomFromString</a>.</p> <p><i>initialRect</i></p> <p>Pointer to a <a href="#">ASFixedRect</a> specifying the destination rectangle, specified in user space coordinates. The appropriate information will be extracted from <i>initialRect</i>, depending on <i>initialFitType</i>, to create the destination. All four of <i>initialRect</i>'s components should be set.</p> <p><i>initialZoom</i></p> <p>The zoom factor to set for the destination. Used only if <i>initialFitType</i> is XYZ. Use the predefined value <i>PDViewDestNULL</i> (see <i>PDExpT.h</i>) to indicate a <i>NULL</i> zoom factor, described in Section 6.8 in the <a href="#">Portable Document Format Reference Manual</a>.</p> <p><i>pageNum</i></p> <p>Currently unused.</p> |
| <b>Return Value</b>    | The newly-created view destination.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b>Related Methods</b> | <a href="#">PDViewDestFromCosObj</a><br><a href="#">PDViewDestDestroy</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

## Example

```
dstPage = PDDocAcquirePage(pdDoc, (offset +
 Pages[i].pg)-1);
dest = PDViewDestCreate(pdDoc, dstPage,
 fit, &initRect, zoom, Pages[i].pg-1);
if(PDViewDestIsValid(dest))
{
 pdAction = PDActionNewFromDest(pdDoc,
 dest, pdDoc);
 PDBookmarkSetAction(newBm, pdAction);
}
```

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDViewDestDestroy

```
void PDViewDestDestroy (PDViewDestination dest);
```

**Description** Deletes a view destination object. Before deleting a view destination, ensure that no link or bookmark refers to it.

**Parameters** *dest*  
The view destination to destroy.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDViewDestFromCosObj](#)  
[PDViewDestCreate](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0  |
|-------------|-----------|------|
| Acrobat     | M, W, U   | M, W |
| Reader      | —         | —    |

## PDViewDestFromCosObj

[PDViewDestination](#) PDViewDestFromCosObj ([CosObj](#) obj);

**Description** Converts the specified Cos object to a view destination and verifies that the view destination is valid. This method does not copy the object, but is instead the logical equivalent of a type cast.

**Parameters** *obj*  
Dictionary Cos object to convert to a view destination.

**Return Value** Array Cos object for the view destination.

**Exceptions** Raises [pdErrBadAction](#) if the destination is invalid, as determined by [PDViewDestIsValid](#).

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDViewDestGetCosObj](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDViewDestGetAttr

```
void PDViewDestGetAttr (PDViewDestination dest, ASInt32* pageNum,
 ASAtom* fitType, ASFixedRectP destRect, ASFixed* zoom);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets a view destination's fit type, destination rectangle, and zoom factor. The destination must be represented by an array, which is the case for a <b>GoToR</b> action.                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Parameters</b>      | <p><i>dest</i></p> <p>The view destination whose attributes are obtained.</p> <p><i>pageNum</i></p> <p>(Filled by the method) The page number of the destination's page.</p> <p><i>fitType</i></p> <p>(Filled by the method) Destination fit type. One of the values listed in <a href="#">View Destination Fit Types</a>.</p> <p><i>destRect</i></p> <p>(Filled by the method) Pointer to a <a href="#">ASFixedRect</a> containing the destination's rectangle, specified in user space coordinates.</p> <p><i>zoom</i></p> <p>(Filled by the method) The destination's zoom factor.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Exceptions</b>      | Raises <a href="#">genErrBadParm</a> if the destination is not represented by an array.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Related Methods</b> | <a href="#">PDViewDestCreate</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

## Example

```

PDLINKAnnot pdAnnot;
PDAction oldAction;
PDViewDestination viewDest;
ASInt32 page;
ASAtom fit;
ASFixedRect destRect;
ASFixed zoom;

oldAction = PDLINKAnnotGetAction(pdAnnot);
viewDest = PDActionGetDest(oldAction);
PDViewDestGetAttr(viewDest, &page, &fit,
 &destRect, &zoom);
PDActionDestroy(oldAction);

```

## Availability

Plug-ins: Available if *PL\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDViewDestGetCosObj

[CosObj](#) PDViewDestGetCosObj ([PDViewDestination](#) dest);

|                        |                                                                                                                                                                                                                                                     |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the Cos object corresponding to a view destination and verifies that the view destination is valid. This method does not copy the object, but is instead the logical equivalent of a type cast.                                                |
| <b>Parameters</b>      | <p><i>dest</i></p> <p>The view destination whose Cos object is obtained.</p>                                                                                                                                                                        |
| <b>Return Value</b>    | Array Cos object for the view destination. The contents of the array can be enumerated using <a href="#">CosObjEnum</a> . Returns a <i>NULL</i> Cos object if the view destination is invalid, as determined by <a href="#">PDViewDestIsValid</a> . |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                    |
| <b>Related Methods</b> | <a href="#">CosObjEnum</a><br><a href="#">PDViewDestFromCosObj</a><br><a href="#">PDViewDestIsValid</a>                                                                                                                                             |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                            |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDViewDestIsValid

```
ASBool PDViewDestIsValid (PDViewDestination dest);
```

|                        |                                                                                                                                                                                                                                                                                                             |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Tests whether or not a view destination is valid. This is intended only to ensure that the view destination has not been deleted, not to ensure that all necessary information is present and valid.                                                                                                        |
| <b>Parameters</b>      | <p><i>dest</i></p> <p>The view destination whose validity is determined.</p>                                                                                                                                                                                                                                |
| <b>Return Value</b>    | <i>true</i> if the view destination is valid, <i>false</i> otherwise.                                                                                                                                                                                                                                       |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                        |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                        |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                            |
| <b>Related Methods</b> | None                                                                                                                                                                                                                                                                                                        |
| <b>Example</b>         | <pre>dstPage = PDDocAcquirePage(pdDoc, (offset +     Pages[i].pg)-1); dest = PDViewDestCreate(pdDoc, dstPage,     fit, &amp;initRect, zoom, Pages[i].pg-1); if(PDViewDestIsValid(dest)) {     pdAction = PDActionNewFromDest(pdDoc,         dest, pdDoc);     PDBookmarkSetAction(newBm, pdAction); }</pre> |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                    |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDViewDestResolve

```
PDViewDestination PDViewDestResolve(PDViewDestination dest,
 PDDoc doc);
```

**Description** Resolves a destination. *dest* is the value of the **D** key in an action. It can be a real destination (an array) or a name. If it is a name, look it up in *doc*'s Dests dictionary. The value found there can be a real dest (an array) or a dictionary. If it's a dictionary, look up the **D** key in that dictionary.

**Parameters**

*dest*  
The destination to resolve.

*doc*  
The *PDDoc* that contains the destination.

**Return Value** The resolved view destination.

**Exceptions** Can raise memory, I/O, and parsing exceptions.

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDActionGetDest](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

**PDWord**

## PDWordFilterString

```
ASBool PDWordFilterString (ASUns16* infoArray, char* cNewWord,
 char* cOldWord);
```

### Description

Removes leading and trailing spaces from the specified word. It also removes punctuation and soft hyphens within the word. Wildcard characters ("\*" and "?") are not removed, while "," and "." are removed if and only if they are surrounded by alphanumeric characters. It also converts ligatures to their constituent characters.

The determination of which characters are alphanumeric, wildcard, punctuation, and so forth, is made by the values in *infoArray*.

Although this method seems very similar to [PDWordFilterWord](#), the two methods treat letters and digits slightly differently. [PDWordFilterWord](#) uses the encoding info array but also does a straight character code test for any characters that have not been mapped to anything. It does this to catch letters and digits from non-standard character sets, and is necessary to avoid removing words with non-standard character sets.

*PDWordFilterString*, on the other hand, was designed for known character sets such as WinAnsi and Mac Roman.

For non-Roman character set viewers, this method currently supports only SHIFT-JIS encoding on a Japanese system.

### Parameters

*infoArray*

An array specifying the type of each character in the font. Each entry in this table must be one of the [Character Type Codes](#). If *infoArray* is set to *NULL*, a default table is used. For non-UNIX Roman systems, it is MacRomanEncoding in Mac OS and WinAnsiEncoding in Windows. In UNIX (except HP-UX) Roman systems, it is ISO8859-1 (ISO Latin-1); for HP-UX, it is HP-ROMAN8. See Appendix C in the [Portable Document Format Reference Manual](#) for descriptions of MacRomanEncoding and WinAnsiEncoding.

*cNewWord*

(Filled by the method) The filtered word.

*cOldWord*

The unfiltered word. This value must be passed to the method.

**Return Value** *true* if the string required filtering, *false* if the filtered string is the same as the unfiltered string.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDWordFilterWord](#)

**Example**

```
extern ASUns16 myInfo[];
char newWord[128], oldWord[128];
PDWordFilterString(&myInfo, newWord,
 oldWord);
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDWordFilterWord

```
ASBool PDWordFilterWord (PDWord word, char* buffer,
 ASInt16 bufferLen, ASInt16* newLen);
```

### Description

Removes non-alphanumeric characters from the specified word and also converts ligatures to their constituent characters. The determination of which characters to remove is made by examining the flags in the *outEncInfo* array passed to [PDDocCreateWordFinder](#). As a result, this method is most useful after you have been called with words obtained by calling [PDWordFinderGetNthWord](#), in the callback for [PDWordFinderEnumWords](#), and words in the *xySortTable* returned by [PDWordFinderAcquireWordList](#). See the description of [PDWordFilterString](#) for further information, and for a description of how the two methods differ.

The Acrobat Catalog program uses this method to filter words before indexing them.

This method works with non-Roman systems.

### Parameters

*word*

The *PDWord* to filter.

*buffer*

(Filled by the method) The filtered string.

*bufferLen*

The maximum number of characters that *buffer* can hold.

*newLen*

(Filled by the method) Number of characters actually written into *buffer*.

### Return Value

*true* if the word required filtering, *false* if the filtered string is the same as the unfiltered string.

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

**Related Methods**    [PDWordFilterString](#)  
[PDWordFinderGetNthWord](#)  
[PDWordFinderEnumWords](#)

**Example**

```

ACCB1 ASBool ACCB2
WordCounterProc(PDWordFinder WordFinder,
PDWord word, ASInt32 PageNum,
void* fileNum)
{
PDWordGetNthQuad(word, 0, &quad);
PDWordFilterWord(word, gbuf,
sizeof(gbuf), &len);
}

```

**Availability**    Plug-ins: Available if *PL\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDWordGetAttr

```
ASUns16 PDWordGetAttr (PDWord word);
```

**Description** Gets a bit field containing information on the type of characters in a word. Use [PDWordGetCharacterTypes](#) if you wish to check each character's type individually.

**Parameters** *word*  
The word whose character types are obtained.

**Return Value** A bit field containing information on the type of characters in *word*. The value is a logical OR of the [Word Attributes](#).

*Note: PDWordGetAttr may return an attribute value greater than the maximum of all of the public attributes since there can be private attributes added on. It is recommended to AND the result with the attribute you are interested in.*

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDWordGetCharacterTypes](#)  
[PDWordGetStyleTransition](#)  
[PDWordGetNthCharStyle](#)  
[PDWordFinderEnumWords](#)  
[PDWordFinderGetNthWord](#)

**Example**

```
attr = PDWordGetAttr(myWord) &
 WXE_HAS_TRAILING_PUNC;
if(attr)
{
 /* process words with trailing punc. */
 ...
}
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |



| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDWordGetCharacterTypes

```
void PDWordGetCharacterTypes (PDWord word, ASUns16* cArr,
 ASInt16 size);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the character type for each character in a word.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>Parameters</b>      | <p><i>word</i></p> <p>The word whose character types are obtained.</p> <p><i>cArr</i></p> <p><i>(Filled by the method)</i> An array of character types. This array contains one element for each character in the word. Use <a href="#">PDWordGetLength</a> to determine the number of elements that must be in the array. Each element is the logical OR of one or more of the <a href="#">Character Type Codes</a>.</p> <p>For non-Roman character set viewers, meaningful values are returned only for Roman characters. For non-Roman characters, it returns 0, which is the same as <i>W_CNTRL</i>. If the character is 2 bytes, both bytes indicate the same character type.</p> <p><i>size</i></p> <p>Number of elements in <i>cArr</i>.</p> |
| <b>Return Value</b>    | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>Related Methods</b> | <a href="#">PDWordGetAttr</a><br><a href="#">PDWordGetLength</a><br><a href="#">PDWordFinderEnumWords</a><br><a href="#">PDWordFinderGetNthWord</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

## Example

```
ASUns16 ctTab[255];

attr = PDWordGetAttr(myWord);
wlen = PDWordGetLength(myWord);
if(attr & WXE_HAS_DIGIT){
 PDWordGetCharacterTypes(myWord, ctTab,
 (ASUns16)wlen);
 IsNumber = false;
 for(i=0;i<wlen;i++){
 if(!(ctTab[i] & W_DIGIT))
 break;
 }
 if(i == wlen)
 IsNumber = true;
}
```

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDWordGetCharDelta

```
ASUns8 PDWordGetCharDelta (PDWord word);
```

**Description** Gets the difference between the word length (the number of printed characters in the word) and the PDF word length (the number of character codes in the word). For instance, if the PDF word is “*fi* (ligature) *sh*” the mapped word will be “fish”. The ligature occupies only one character code, so in this case the character delta will be  $3-4 = -1$ .

**Parameters** *word*  
The word whose character delta is obtained.

**Return Value** The character delta for *word*.  
If the [PDWord](#)'s character set has no ligatures, such as on a non-Roman viewer supporting Japanese, returns 0.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDWordGetLength](#)  
[PDWordGetCharOffset](#)  
[PDWordFinderEnumWords](#)  
[PDWordFinderGetNthWord](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDWordGetCharOffset

```
ASUns16 PDWordGetCharOffset (PDWord word);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Returns a word's character offset from the beginning of its page. This information, together with the character delta obtained from <a href="#">PDWordGetCharDelta</a> , can be used to highlight a range of words on a page, using <a href="#">PDTextSelectCreatePageHilite</a> .                                                                                                                                                                                                                         |
| <b>Parameters</b>      | <p><i>word</i></p> <p>The word whose character offset is obtained.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Return Value</b>    | The word's character offset. On multibyte systems, it points to the first byte.                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Related Methods</b> | <a href="#">PDWordGetCharDelta</a><br><a href="#">PDWordGetLength</a><br><a href="#">PDTextSelectCreatePageHilite</a><br><a href="#">PDWordFinderEnumWords</a><br><a href="#">PDWordFinderGetNthWord</a>                                                                                                                                                                                                                                                                                                   |
| <b>Example</b>         | <pre>ACCB1 ASBool ACCB2 WordCounterProc(     PDWordFinder WordFinder, PDWord word,     ASInt32 PageNum, void* fileNum) {     ASUns32 cur;      cur = (ASUns32) PDWordGetCharOffset(word);     WordCnt += (cur - gLast);     gLast = cur; }  WordFinder = PDDocCreateWordFinder(     CurrentPDDoc, NULL, NULL, NULL, 0,     WXE_PDF_ORDER, NULL); PDWordFinderEnumWords(WordFinder, PageNum,     ASCallbackCreateProto(PDWordProc,     &amp;WordCounterProc), NULL); PDWordFinderDestroy(WordFinder);</pre> |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                                                                                                                                                                   |

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDWordGetLength

```
ASUns8 PDWordGetLength (PDWord word);
```

**Description** Gets the number of characters in a word. This method also works on non-Roman systems.

**Parameters** *word*  
The word object whose character count is obtained.

**Return Value** The number of characters in *word*.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDWordGetCharDelta](#)  
[PDWordGetCharOffset](#)  
[PDWordFinderEnumWords](#)  
[PDWordFinderGetNthWord](#)

**Example**

```
ASUns16 ctTab[255];
attr = PDWordGetAttr(myWord);
wlen = PDWordGetLength(myWord);
if(attr & WXE_HAS_DIGIT)
{
 PDWordGetCharacterTypes(myWord, ctTab,
 (ASUns16)wlen);
 IsNumber = false;
 for(i=0;i<wlen;i++)
 {
 if(!(ctTab[i] & W_DIGIT))
 break;
 }
 if(i == wlen)
 IsNumber = true;
}
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |



## PDWordGetNthCharStyle

[PDStyle](#) PDWordGetNthCharStyle ([PDWordFinder](#) wObj, [PDWord](#) word, ASInt32 dex);

|                        |                                                                                                                                                                                                                                          |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Returns a <i>PDStyle</i> object for the $n^{\text{th}}$ style in a word.                                                                                                                                                                 |
| <b>Parameters</b>      | <p><i>wObj</i><br/>A word finder object.</p> <p><i>word</i><br/>The word whose <math>n^{\text{th}}</math> style is obtained.</p> <p><i>dex</i><br/>The index of the style to obtain. The first style in a word has an index of zero.</p> |
| <b>Return Value</b>    | The $n^{\text{th}}$ style in the word. Returns <i>NULL</i> if <i>dex</i> is greater than the number of styles in the word.                                                                                                               |
| <b>Exceptions</b>      | Raises <a href="#">genErrBadParm</a> if <i>dex</i> < 0.                                                                                                                                                                                  |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                         |
| <b>Related Methods</b> | <a href="#">PDWordGetStyleTransition</a><br><a href="#">PDDocGetWordFinder</a>                                                                                                                                                           |
| <b>Example</b>         | <pre>for(pdwPtr = rdOrderTable;size;size--) {     myWord = *pdwPtr++;     PDWordGetNthQuad(myWord, 0, &amp;quad);     style = PDWordGetNthCharStyle(pdWF,         myWord, 0); }</pre>                                                    |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                 |

### Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |      |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0  |
| Acrobat     | M, W, U   | M, W |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Reader      | M, W, U   | M, W, U |

## PDWordGetNthQuad

```
ASBool PDWordGetNthQuad (PDWord word, ASInt16 nTh,
 ASFixedQuad* quad);
```

### Description

Gets the specified word's  $n^{\text{th}}$  quad, specified in user space coordinates. See [PDWordGetNumQuads](#) for a description of a quad.

The quad's height is the height of the font's bounding box, not the height of the tallest character used in the word. The font's bounding box is determined by the glyphs in the font that extend farthest above and below the baseline; it often extends somewhat above the top of "A" and below the bottom of "y."

The quad's width is determined from the characters actually present in the word.

As an example, the quads for the words "AWAY" and "away" have the same height, but generally do not have the same width unless the font is a monospaced font (a font in which all characters have the same width).

### Parameters

*word*

The word whose  $n^{\text{th}}$  quad is obtained.

*nTh*

The quad to obtain. A word's first quad has an index of zero.

*quad*

*(Filled by the method)* Pointer to the word's  $n^{\text{th}}$  quad, specified in user space coordinates.

### Return Value

*true* if the word has an  $n^{\text{th}}$  quad, *false* otherwise.

### Exceptions

None

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDWordGetNumQuads](#)

## Example

```
ACCB1 ASBool ACCB2 WordCounterProc(
 PDWordFinder WordFinder, PDWord word,
 ASInt32 PageNum, void* fileNum)
{
 PDWordGetNthQuad(word, 0, &quad);
 PDWordFilterWord(word, gbuf,
 sizeof(gbuf), &len);
}
WordFinder = PDDocCreateWordFinder(
 CurrentPDDoc, NULL, NULL, NULL, 0,
 WXE_PDF_ORDER, NULL);
PDWordFinderEnumWords(WordFinder, PageNum,
 ASCallbackCreateProto(PDWordProc,
 &WordCounterProc), NULL);
PDWordFinderDestroy(WordFinder);
```

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDWordGetNumQuads

```
ASInt16 PDWordGetNumQuads (PDWord word);
```

**Description** Gets the number of quads in a word. A quad is a quadrilateral bounding a contiguous piece of a word. Every word has at least one quad. A word has more than one quad, for example, if it is hyphenated and split across multiple lines or if the word is set on a curve rather than on a straight line.

**Parameters** *word*  
The word whose quad count is obtained.

**Return Value** The number of quads in *word*.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDWordGetNthQuad](#)

**Example**

```
for(pdwPtr = rdOrderTable;size;size--){
 myWord = *pdwPtr++;
 if(PDWordGetNumQuads(myWord) > 1)
 continue;
 PDWordGetNthQuad(myWord, 0, &quad);
 style = PDWordGetNthCharStyle(pdWF,
 myWord, 0);
}
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDWordGetString

```
void PDWordGetString (PDWord word, char* str, ASInt32 len);
```

### Description

Gets a word's text and also converts ligatures to their constituent characters. The string to return includes any word break characters (such as space characters) that follow the word, but not any that precede the word. The characters that are treated as word breaks are defined in the *outEncInfo* parameter of [PDDocCreateWordFinder](#) method. Use [PDWordFilterString](#) to subsequently remove the word break characters. This method does *not* convert ligatures to their constituent characters, so the character codes for the ligature remain in the string. This method produces a string in whatever encoding the [PDWord](#) uses, for both Roman and non-Roman systems.

### Parameters

*word*

The word whose string is obtained.

*str*

(Filled by the method) The string. The encoding of the string is the encoding used by the [PDWordFinder](#) that supplied the [PDWord](#). For instance, if [PDDocCreateWordFinderUCS](#) is used to create the word finder, [PDWordGetString](#) returns only Unicode.

*Note:* There is no way to detect Unicode strings returned by [PDWordGetString](#), since there is no UCS header (FEFF) added to each string returned.

*len*

Length of *str*, in bytes. Up to *len* characters of *word* will be copied into *str*. If *str* is long enough, it will be *NULL*-terminated.

### Return Value

None

### Exceptions

Raises [genErrBadParm](#) if either *word* or *str* is *NULL*.

### Notifications

None

### Header File

*PDCalls.h*

**Related Methods**    [PDWordGetLength](#)  
[PDWordGetCharDelta](#)  
[PDWordFinderEnumWords](#)  
[PDWordFinderGetNthWord](#)

**Example**            `char buf[128];`

```

PDWordGetString(myWord, buf, sizeof(buf));
attr = PDWordGetAttr(myWord);
if(attr & WXE_HAS_TRAILING_PUNC){
 buf[PDWordGetLength(myWord)-1] = 0x00;
}

```

**Availability**        Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDWordGetStyleTransition

```
ASInt16 PDWordGetStyleTransition (PDWord word,
 ASInt16* transTbl, ASInt16 size);
```

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b>     | Gets the locations of style transitions in a word. Every word has at least one style transition, at character position zero in the word.                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Parameters</b>      | <p><i>word</i></p> <p>The word whose style transition list is obtained.</p> <p><i>transTbl</i></p> <p>(Filled by the method) Array of style transitions. Each element is the character offset in <i>word</i> where the style changes. The offset specifies the first character in the word that has the new style. The first character in a word has an offset of zero.</p> <p><i>size</i></p> <p>Number of entries that <i>transTbl</i> can hold. The word is searched only until this number of style transitions have been found.</p> |
| <b>Return Value</b>    | Number of style transition offsets copied to <i>transTbl</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Exceptions</b>      | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Notifications</b>   | None                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>Related Methods</b> | <a href="#">PDWordGetNthCharStyle</a><br><a href="#">PDWordFinderEnumWords</a><br><a href="#">PDWordFinderGetNthWord</a>                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Availability</b>    | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i> ) is set to 0x00020000 or higher.                                                                                                                                                                                                                                                                                                                                                                                                                                 |

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |



|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Reader      | M, W, U   | M, W, U |

## PDWordIsRotated

```
ASBool PDWordIsRotated (PDWord word);
```

|                        |                                                                                 |
|------------------------|---------------------------------------------------------------------------------|
| <b>Description</b>     | Tests whether or not a word is rotated.                                         |
| <b>Parameters</b>      | <i>word</i><br>The word to test.                                                |
| <b>Return Value</b>    | <i>true</i> if the word is rotated, <i>false</i> otherwise.                     |
| <b>Exceptions</b>      | None                                                                            |
| <b>Notifications</b>   | None                                                                            |
| <b>Header File</b>     | <i>PDCalls.h</i>                                                                |
| <b>Related Methods</b> | <a href="#">PDWordFinderEnumWords</a><br><a href="#">PDWordFinderGetNthWord</a> |

**Example**

```
for(pdwPtr = xySortTable;numWords;
 numWords--)
{
 myWord = *pdwPtr++;
 if(PDWordIsRotated(myWord))
 break;
}
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

### Available in:

|                   |           |         |
|-------------------|-----------|---------|
| Library API       | 1.0       | 4.0     |
| Adobe PDF Library | M, W, U   | M, W, U |
| Plug-in API       | 3.0, 3.01 | 4.0     |
| Acrobat           | M, W, U   | M, W    |
| Reader            | M, W, U   | M, W, U |

## PDWordSplitString

```
ASUns16 PDWordSplitString (ASUns16* infoArray, char* cNewWord,
 char* cOldWord, ASUns16 nMaxLen);
```

### Description

Splits the specified string into words by substituting spaces for word separator characters. The list of characters considered to be word separators can be specified, or a default list can be used.

"," and "." are context-sensitive word separator characters. If a "," or a "." are surrounded by digits (for example, 654,096.345), they are not considered a word separators.

For non-Roman character set viewers, this method currently supports only SHIFT-JIS encoding on a Japanese system.

### Parameters

*infoArray*

A character information table. It specifies each character's type; word separator characters must be marked as *W\_WORD\_BREAK* (see [Character Type Codes](#)). This table can be identical to the table to pass to [PDDocCreateWordFinder](#). If *infoArray* is *NULL*, a default table is used (see [Glyph Names of Word Separators](#)).

*cNewWord*

(Filled by the method) Word that has been split. Word separator characters have been replaced with spaces.

*cOldWord*

Word to split.

*nMaxLen*

Number of characters that *cNewWord* can hold. Word splitting stops when *cOldWord* is completely processed or *nMaxLen* characters have been placed in *cNewWord*, whichever occurs first.

### Return Value

The number of splits that occurred.

### Exceptions

Raises [genErrGeneral](#) if *infoArray* is *NULL*, but host encoding cannot be obtained.

### Notifications

None

**Header File** *PDCalls.h*

**Related Methods** [PDDocCreateWordFinder](#)

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## **PDWordFinder**

## PDWordFinderAcquireWordList

```
void PDWordFinderAcquireWordList (PDWordFinder wObj,
 ASInt32 pgNum, PDWord* wInfoP, PDWord** xySortTable,
 PDWord** rdOrderTable, ASInt32* numWords);
```

### Description

Finds all words on the specified page and returns one or more tables containing the words. One table contains the words sorted in the order in which they appear in the PDF file, while the other contains the words sorted by their *x*- and *y*-coordinates on the page.

Only words within the page's crop box (see [PDPageGetCropBox](#)) are enumerated. Words outside the crop box are skipped.

There can be only one word list in existence at a time; plug-ins must release the previous word list, using [PDWordFinderReleaseWordList](#), before creating a new one.

Use [PDWordFinderEnumWords](#) instead of this method if you wish to find one word at a time instead of obtaining a table containing all words on a page.

### Parameters

*wObj*

The word finder (created using [PDDocCreateWordFinder](#) or [PDDocCreateWordFinderUCS](#)) used to acquire the word list.

*pgNum*

The page number for which words are found.

*wInfoP*

*(Filled by the method)* Pointer to an array of words found. The words are ordered in PDF order; that is, the order in which they appear in the PDF file's data. This is often, but not always, the order in which a person would read the words. Use [PDWordFinderGetNthWord](#) to traverse this array. This array is always filled, regardless of the flags used in the call to [PDDocCreateWordFinder](#) or [PDDocCreateWordFinderUCS](#).

## *xySortTable*

(Filled by the method) Pointer to an array of indices into *wInfoP*, sorted in *x-y* order; that is, all words on the first "line," from left to right, followed by all words on the next "line." This array is only filled if the *WXE\_XY\_SORT* flag was set in the call to [PDDocCreateWordFinder](#) or [PDDocCreateWordFinderUCS](#).

## *rdOrderTable*

Currently unused.

## *numWords*

(Filled by the method) The number of words found on the page.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocCreateWordFinder](#)  
[PDDocCreateWordFinderUCS](#)  
[PDWordFinderReleaseWordList](#)  
[PDWordFinderEnumWords](#)  
[PDWordFinderGetNthWord](#)

## Example

```
PDWordFinder pdWF =
 PDDocCreateWordFinder(pdDoc, NULL,
 NULL, NULL, 0, WXE_RD_ORDER_SORT, NULL);
tmp = PDDocAcquirePage(pdDoc, page);

PDWordFinderAcquireWordList(pdWF, page,
 &wInf, NULL, &rdOrderTable, &size);
if(size == 0){
 PDPageRelease(tmp);
 PDWordFinderDestroy(pdWF);
 E_RTRN_VOID
}
if(rdOrderTable){
 for(pdwPtr = rdOrderTable;size;size--){
 myWord = *pdwPtr++;
 PDWordGetNthQuad(myWord, 0, &quad);
 style = PDWordGetNthCharStyle(pdWF,
 myWord, 0);
 font = PDStyleGetFont(style);
 PDFontGetName(font, mbuf,
 sizeof(mbuf));
 if(strstr(mbuf, "Bold"))
 IsBold = TRUE;
 else
 IsBold = FALSE;
 if(IsBold){
 PDWordGetString(myWord, subtitle,
 sizeof(subtitle));
 if (strstr(title, subtitle)){
 QuadToRect(&quad, &destRect);
 break;
 }
 }
 }
 PDWordFinderReleaseWordList(pdWF, page);
}
newview = PDViewDestCreate(pdDoc, tmp,
 k_XYZ, &destRect, fixedZero, page);
PDWordFinderDestroy(pdWF);
```

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:



|                   |         |         |
|-------------------|---------|---------|
| Library API       | 1.0     | 4.0     |
| Adobe PDF Library | M, W, U | M, W, U |

|             |           |         |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0     |
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDWordFinderDestroy

```
void PDWordFinderDestroy (PDWordFinder wObj);
```

**Description** Destroys a word finder. Use this when you are done extracting text in a file.

**Parameters** *wObj*  
The word finder to destroy.

**Return Value** None

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocCreateWordFinder](#)  
[PDDocCreateWordFinderUCS](#)  
[PDDocGetWordFinder](#)

**Example**

```
ACCB1 ASBool ACCB2
WordCounterProc(
 PDWordFinder WordFinder, PDWord word,
 ASInt32 PageNum, void *fileNum)
{
 ASUns32 cur;
 cur = (ASUns32)
 PDWordGetCharOffset(word);
 WordCnt += (cur - gLast);
 gLast = cur;
}
WordFinder = PDDocCreateWordFinder(
 CurrentPDDoc, NULL, NULL, NULL, 0,
 WXE_PDF_ORDER, NULL);
PDWordFinderEnumWords(WordFinder, PageNum,
 ASCallbackCreateProto(PDWordProc,
 &WordCounterProc), NULL);
PDWordFinderDestroy(WordFinder);
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDWordFinderEnumWords

```
ASBool PDWordFinderEnumWords (PDWordFinder wObj,
 ASInt32 pageNum, PDWordProc wordProc, void* clientData);
```

### Description

Extracts words, one at a time, from the specified page or the entire document. Calls a user-supplied procedure once for each word found. If you wish to extract all text from a page at once, use [PDWordFinderAcquireWordList](#) instead of this method.

Only words within the page's crop box (see [PDPageGetCropBox](#)) are enumerated. Words outside the crop box are skipped.

### Parameters

*wObj*

A word finder object.

*pageNum*

Page number from which to extract words. Pass *PDAllPages* (see *PDExpT.h*) to sequentially process all pages in the document.

*wordProc*

User-supplied callback to call once for each word found. Enumeration halts if *wordProc* returns *false*.

*clientData*

Pointer to user-supplied data to pass to *wordProc* each time it is called.

### Return Value

*true* if enumeration was successfully completed.

*false* if enumeration was terminated because *wordProc* returned *false*.

### Exceptions

Raises [genErrBadParm](#) if *wordProc* is *NULL*, or *pageNum* is less than zero or greater than the total number of pages in the document.

### Notifications

None

### Header File

*PDCalls.h*

### Related Methods

[PDDocCreateWordFinder](#)  
[PDDocCreateWordFinderUCS](#)  
[PDDocGetWordFinder](#)  
[PDWordFinderAcquireWordList](#)

## Example

```
ACCB1 ASBool ACCB2 WordCounterProc(
 PDWordFinder WordFinder, PDWord word,
 ASInt32 PageNum, void* fileNum)
{
 ASUns32 cur;

 cur = (ASUns32) PDWordGetCharOffset(word);
 WordCnt += (cur - gLast);
 gLast = cur;
}
WordFinder = PDDocCreateWordFinder(
 CurrentPDDoc, NULL, NULL, NULL, 0,
 WXE_PDF_ORDER, NULL);
PDWordFinderEnumWords(WordFinder, PageNum,
 ASCallbackCreateProto(PDWordProc,
 &WordCounterProc), NULL);
PDWordFinderDestroy(WordFinder);
```

## Availability

Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

## Available in:

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDWordFinderGetLatestAlgVersion

```
ASInt16 PDWordFinderGetLatestAlgVersion (PDWordFinder wObj);
```

**Description** Gets the version number of the specified word finder, or the version number of the latest word finder algorithm.

**Parameters** *wObj*  
The word finder whose algorithm's version is obtained. Pass *NULL* to obtain the latest word finding algorithm version number.

**Return Value** The algorithm version associated with *wObj*, or the version of the latest word finder algorithm if *wObj* is *NULL*.

**Exceptions** None

**Notifications** None

**Header File** *PDCalls.h*

**Related Methods** [PDDocCreateWordFinder](#)  
[PDDocCreateWordFinderUCS](#)  
[PDDocGetWordFinder](#)

**Example**

```
/* store this in our index file so we can
 subsequently highlight text using the
 correct algorithm version */
ASInt16 version =
 PDWordFinderGetLatestAlgVersion(
 WordFinder);
```

**Availability** Plug-ins: Available if *PI\_PDMODEL\_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

**Available in:**

| Library API       | 1.0     | 4.0     |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0     |
|-------------|-----------|---------|
| Acrobat     | M, W, U   | M, W    |
| Reader      | M, W, U   | M, W, U |

## PDWordFinderGetNthWord

[PDWord](#) PDWordFinderGetNthWord ([PDWordFinder](#) wObj, ASInt32 nTh);

- Description** Gets the  $n^{\text{th}}$  word in the word list obtained using [PDWordFinderAcquireWordList](#).
- Parameters**
- wObj*  
The word finder whose  $n^{\text{th}}$  word is obtained.
- nTh*  
The index of the word to obtain. The first word on a page has an index of zero. Words are counted in PDF order. See the description of the *wInfoP* parameter in [PDWordFinderAcquireWordList](#).
- Return Value** The  $n^{\text{th}}$  word. Returns *NULL* when the end of the list is reached.
- Exceptions** None
- Notifications** None
- Header File** *PDCalls.h*
- Related Methods** [PDWordFinderAcquireWordList](#)  
[PDWordFinderEnumWords](#)
- Example**
- ```
PDWord word =
    PDWordFinderGetNthWord(WordFinder, 0);
```
- Availability** Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDWordFinderReleaseWordList

```
void PDWordFinderReleaseWordList (PDWordFinder wObj,
    ASInt32 pageNum);
```

| | |
|------------------------|--|
| Description | Releases the word list for a given page. Use this to release a list created by PDWordFinderAcquireWordList when you are done using this list. |
| Parameters | <p><i>wObj</i> A word finder object.</p> <p><i>pageNum</i> Number of pages for which a word list is released.</p> |
| Return Value | None |
| Exceptions | Raises genErrBadUnlock if the list has already been released. |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | PDWordFinderAcquireWordList PDDocCreateWordFinder PDDocCreateWordFinderUCS PDDocGetWordFinder |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDXObject

PDXObjectEnumFilters

```
void PDXObjectEnumFilters (PDXObject obj,
    PDXObjectFilterEnumProc proc, void* clientData);
```

Description Enumerates the filters attached to an *XObject*, calling a user-supplied procedure for each filter.

Parameters

obj

The *XObject* whose filters are enumerated.

proc

User-supplied callback to call for each filter attached to the *XObject*.

Enumeration ends if *proc* returns *false*. *proc* will not be called if there are no filters attached to the *XObject*.

clientData

Pointer to user-supplied data to pass to *proc* each time it is called.

Return Value None

Exceptions None

Notifications None

Header File *PDCalls.h*

Related Methods None

Availability Plug-ins: Available if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDXObjectGetCosObj

CosObj PDXObjectGetCosObj (**PDXObject** xObj);

| | |
|------------------------|---|
| Description | Gets the Cos object associated with an <i>XObject</i> . This method does not copy the object, but is instead the logical equivalent of a type cast. |
| Parameters | <i>xObj</i> The <i>XObject</i> whose Cos object is obtained. |
| Return Value | The dictionary Cos object for the <i>XObject</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDXObjectGetData

```
void PDXObjectGetData (PDXObject obj, PDGetDataProc getDataProc,
    void* clientData);
```

| | |
|------------------------|--|
| Description | Passes the data from an <i>XObject</i> to a user-supplied procedure. |
| Parameters | <p><i>obj</i></p> <p>The <i>XObject</i> whose data is read.</p> <p><i>getDataProc</i></p> <p>User-supplied callback to call with the <i>XObject</i>'s data.</p> <p>Enumeration ends if <i>getDataProc</i> returns <i>false</i>.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>getDataProc</i>.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDXObjectGetDataLength

```
ASInt32 PDXObjectGetDataLength (PDXObject xObj);
```

| | |
|------------------------|---|
| Description | Gets the value of the <i>XObject</i> stream's <i>/Length</i> key, which specifies the amount of data in the PDF file (that is, after all compression/encoding filters have been applied). |
| Parameters | <p><i>xObj</i></p> <p>The <i>XObject</i> whose data length is obtained.</p> |
| Return Value | The <i>XObject</i> 's data length. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDXObjectGetSubtype

[ASAtom](#) PDXObjectGetSubtype ([PDXObject](#) xObj);

| | |
|------------------------|---|
| Description | Gets the subtype of an <i>XObject</i> . Examples of subtype are Image and Form. |
| Parameters | <i>xObj</i> The <i>XObject</i> whose subtype is obtained. |
| Return Value | The <i>ASAtom</i> corresponding to the <i>XObject</i> 's subtype. Can be converted into a string using ASAtomGetCount . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M, W, U | M, W |
| Reader | M, W, U | M, W, U |

PDFEdit

Dump

PDELogDump

```
void PDELogDump ( PDEObjectDumpProc proc, void* clientData);
```

| | |
|------------------------|--|
| Description | Enumerates the internal log of PDEObject s. This is useful when looking for orphaned objects. |
| Parameters | <p><i>proc</i></p> <p>Callback to call once for each PDEObject.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>proc</i> each time it is called.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType peErrUnknownPDEColorSpace genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEObjectDump |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEObjectDump

```
void PDEObjectDump (PDEObject obj, ASInt32 levels,
PDEObjectDumpProc proc, void* clientData);
```

Description

Dumps an ASCII version of an object, its children, and their attributes. The dump contains information about each individual object, separated by spaces, and information for each separate object, separated by tabs. The information for each object is:

- char* — String describing Object Type. (See [PDEObjectGetType](#).)
- long — Number representing Object Type. (See *PEExpt.h* *PDEType* enum.)
- int — The object reference count.
- int — The memory location for the object.

Parameters

obj

The *PDEObject* to dump.

levels

Depth of children and attributes to dump.

proc

Callback for the dump information. It may be called more than once per object.

clientData

Pointer to user-supplied data to pass to *proc* each time it is called.

Read/Write

Read

Return Value

None

Exceptions

[genErrBadParm](#)

Notifications

None

Header File

PERCalls.h

Related Methods

[PDELogDump](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEAttrEnumTable

```
void PDEAttrEnumTable (PDEAttrEnumProc enumProc,
    void* clientData);
```

| | |
|------------------------|---|
| Description | Enumerates the table of attributes. This method enumerates the shared resource objects. It is useful when looking for orphaned attributes. |
| Parameters | <p><i>enumProc</i></p> <p>Callback to call for each attribute.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data to pass to <i>enumProc</i> each time it is called.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

General

PDEDefaultGState

```
void PDEDefaultGState (PDEGraphicStateP stateP,  
    ASInt32 stateSize);
```

| | |
|------------------------|--|
| Description | Fills out a PDEGraphicState structure with the default graphic state. |
| Parameters | <p><i>stateP</i></p> <p>(Filled by the method) Pointer to a PDEGraphicState structure with the default graphic state.</p> <p><i>stateSize</i></p> <p>Size of PDEGraphicState , in bytes.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEEnumElements

```
void PDEEnumElements (const CosObj* contents,
    const CosObj* resources, ASUns32 flags,
    PDEElementEnumProc enumProc, void* enumProcClientData);
```

Description

Enumerates all the *PDEElements* in a given stream. Similar to [PDEContentCreateFromCosObj](#), but provides enumeration instead of a list of elements.

If Marked Content is not ignored, each [PDEContainer](#) contains a [PDEContent](#) list within itself.

Parameters

contents

Cos object that is the source for the content stream. May be page contents, a Form *XObject*, or a Type 3 font CharProc.

resources

The object's Resources dictionary. If the Form or Type 3 font contains a Resources dictionary, this dictionary must be passed in *resources*. Otherwise, it must be the page resources object of the page containing *contents*.

flags

Flags from [PDEEnumElementsFlags](#).

enumProc

User-supplied callback to call once for each top-level element.

enumProcClientData

Pointer to user-supplied data to pass to *enumProc* each time it is called.

Read/Write

Read

Return Value

None

Exceptions

[peErrWrongPDEObjectType](#)
[genErrBadParm](#)
[peErrPStackUnderflow](#)
[peErrCantGetImageDict](#)

Notifications

None

Header File

PERCalls.h

Related Methods [PDEContentCreateFromCosObj](#)

Availability Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEMergeResourcesDict

```
void PDEMergeResourcesDict (CosObj* resDictP, CosDoc cosDoc,
    const CosObj* newResP);
```

Description

Merges two Resources dictionaries in the *same CosDoc*—you *cannot* merge two resource dictionaries from different *CosDocs*.

Both dictionaries and what they reference must be in *cosDoc*. The objects referenced by *newResP* are appended to *resDictP*.

This method only operates on the Cos dictionaries. It assumes there are no resource name conflicts.

Note: Since PDFEdit resolves resource names across [PDEContent](#) objects, this routine is safe for using with PDFEdit methods. This method may be unsafe if you modify streams and dictionaries outside of the PDFEdit API.

This method is useful for adding form resources to page resource dictionaries. This is only necessary if creating PDF 1.1 or earlier files for use with Acrobat 2.1 or earlier. This is unnecessary if creating PDF 1.2 documents.

Parameters

resDictP

(Filled by the method) Dictionary to which *newResP* is merged. When the method completes, *resDictP* is the merged dictionary result.

cosDoc

CosDoc containing both dictionaries.

newResP

Dictionary to merge with *resDictP*.

Read/Write

Write

Return Value

None

Exceptions

[genErrBadParm](#)

Notifications

None

Header File

PEWCalls.h

Related Methods

None

Availability

Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEPurgeCache

```
void PDEPurgeCache (PDDoc doc);
```

Description Clears the PDE Cache of this *PDDoc*. This method is only of interest to plug-ins.

Note: We do not recommend calling this method directly; it is provided only for backwards compatibility.

Parameters *doc*
A *PDDoc* whose cache is purged.

Return Value None

Exceptions None

Notifications None

Header File *PEWCalls.h*

Related Methods None

Availability Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEClip

PDEClipAddElem

```
void PDEClipAddElem (PDEClip clip, ASInt32 addAfterIndex,
PDEElement pdeElement);
```

Description Adds an element to a clip path. The element may be a [PDEPath](#), a [PDEText](#), a [PDEContainer](#), a [PDEGroup](#), or a [PDEPlace](#).

Parameters

clip
The clip path to which an element is added.

addAfterIndex
The index after which to add *pdeElement*. Use *kPDEBeforeFirst* to insert an element at the beginning of the clip object.

pdeElement
The element added, which may be a *PDEGroup*, *PDEPath*, or a *PDEText* object.

Read/Write Write

Return Value None

Exceptions [peErrWrongPDEObjectType](#)
[genErrBadParm](#)

Notifications None

Header File *PEWCalls.h*

Related Methods [PDEClipRemoveElems](#)

Availability Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEClipCopy

```
PDEClip PDEClipCopy (PDEClip srcClip);
```

| | |
|------------------------|--|
| Description | Makes a deep copy of a <i>PDEClip</i> object. |
| Parameters | <i>srcClip</i> The clipping path to copy. |
| Read/Write | Write |
| Return Value | The deep copy of <i>srcClip</i> . |
| Exceptions | Raises if unable to allocate memory. |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEClipCreate

PDEClip PDEClipCreate (void);

| | |
|------------------------|--|
| Description | Creates an empty clip object. |
| Parameters | None |
| Read/Write | Write |
| Return Value | The newly-created clip object. |
| Exceptions | None |
| Notifications | Raises if unable to allocate memory. |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEClipFlattenedEnumElems

```
ASBool PDEClipFlattenedEnumElems (PDEClip clip,  
    PDEClipEnumProc enumProc, void* enumProcClientData);
```

Description For a given *PDEClip*, enumerates all of the *PDEElements* in a flattened manner. In other words, *PDEContainers* and *PDEGroups* nested in the *PDEClip* will not be handed back, but any *PDEPaths* and *PDETexts* nested in them will be. Additionally, *PDEPlace* objects inside the *PDEClip* are not returned.

Parameters

clip
The *PDEClip* to enumerate.

enumProc
Called with each flattened element.
Enumeration continues until all elements have been enumerated, or until *enumProc* returns *false*.

enumProcClientData
Pointer to user-supplied data to pass to *enumProc* each time it is called.

Return Value Returns value of *enumProc*. *true* if successful, *false* otherwise.

Exceptions [peErrWrongPDEObjectType](#)
[genErrBadParm](#)

Notifications None

Header File *PERCalls.h*

Related Methods [PDEClipCreate](#)
[PDEClipGetElem](#)
[PDEClipGetNumElems](#)
[PDEClipRemoveElems](#)

Availability Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|---------|
| Adobe PDF Library | — | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDEClipGetElem

[PDEElement](#) PDEClipGetElem ([PDEClip](#) clip, ASInt32 index);

| | |
|------------------------|---|
| Description | Gets an element from a clip object. |
| Parameters | <p><i>clip</i></p> <p>The clip object from which an element is obtained.</p> <p><i>index</i></p> <p>Index of element to get from <i>clip</i>.</p> |
| Read/Write | Read |
| Return Value | The element from the clip object. |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEClipGetNumElems |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEClipGetNumElems

```
ASInt32 PDEClipGetNumElems (PDEClip clip);
```

| | |
|------------------------|--|
| Description | Gets the number of path and charpath elements in a clip object. Paths are represented as <i>PDEPath</i> objects; charpaths are represented as <i>PDEText</i> objects. |
| Parameters | <i>clip</i> The clip object to examine. |
| Read/Write | Read |
| Return Value | Number of path and charpath elements in <i>clip</i> . If <i>clip</i> contains <i>PDEGroups</i> , this method returns the top-level <i>PDEGroup</i> , <i>PDEPath</i> , or a <i>PDEText</i> object. Use PDEClipFlattenedEnumElems to see only the <i>PDEPath</i> and <i>PDEText</i> objects. |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEClipFlattenedEnumElems PDEClipGetElem |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEClipRemoveElems

```
void PDEClipRemoveElems (PDEClip clip, ASInt32 index,
    ASInt32 count);
```

| | |
|------------------------|--|
| Description | Removes one or more elements from a clip object. |
| Parameters | <p><i>clip</i></p> <p>The clip object from which an element is removed.</p> <p><i>index</i></p> <p>First element to remove.</p> <p><i>count</i></p> <p>Number of elements to remove.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEClipAddElem |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEColorSpace

PDEColorSpaceCreate

```
PDEColorSpace PDEColorSpaceCreate (ASAtom family,
    PDEColorSpaceStruct* csStruct);
```

Description Creates a new color space object of the specified type.

Parameters

family

Supports all PDF 1.3 color spaces, which include:

Device-dependent names: **DeviceCMYK**, **DeviceGray**, **DeviceN**, **DeviceRGB**, or **ICCBased**.

Device-independent names: **CalGray**, **CalRGB**, or **Lab**.

Special names: **Indexed**, **Pattern**, or **Separation**.

Resource names: Arbitrary name specified in page Resources dictionary, including **DefaultGray** or **DefaultRGB**.

csStruct

Data for the type of color space you want to create.

Return Value The newly-created *PDEColorSpace*.

Exceptions

[cosErrExpectedArray](#)
[genErrBadParm](#)
[peErrUnknownPDEColorSpace](#)

Header File *PEWCalls.h*

Related Methods [PDEColorSpaceCreateFromCosObj](#)

Availability Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEColorSpaceCreateFromCosObj

```
PDEColorSpace PDEColorSpaceCreateFromCosObj
(const CosObj* cosObjP);
```

| | |
|------------------------|--|
| Description | Creates a new color space object from a Cos object. Call PDERelease to dispose of the returned color space when finished with it. |
| Parameters | <i>cosObjP</i> Either a <i>CosName</i> for one of the device color spaces: CalGray , CalRGB , Lab , DeviceCMYK , DeviceGray , DeviceN , DeviceRGB , or ICCBased , or a <i>CosArray</i> with the first element a <i>CosName</i> for one of the special color spaces: Indexed , Pattern , or Separation with the second element the <i>CosName</i> for the base color space. |
| Read/Write | Write |
| Return Value | The newly-created <i>PDEColorSpace</i> . |
| Exceptions | cosErrExpectedArray genErrBadParm peErrUnknownPDEColorSpace |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEColorSpaceCreateFromName |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEColorSpaceCreateFromName

[PDEColorSpace](#) PDEColorSpaceCreateFromName ([ASAtom](#) name);

| | |
|------------------------|--|
| Description | Creates a new color space object. Call PDERelease to dispose of the returned color space when finished with it. |
| Parameters | <i>name</i> The <i>ASAtom</i> for the name of the color space created. The name must be one of the following: DeviceCMYK , DeviceGray , or DeviceRGB . |
| Read/Write | Write |
| Return Value | The newly-created <i>PDEColorSpace</i> . |
| Exceptions | cosErrExpectedName genErrBadParm peErrUnknownPDEColorSpace |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEColorSpaceCreate |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEColorSpaceGetBase

```
ASAtom PDEColorSpaceGetBase (PDEColorSpace colorSpace);
```

| | |
|------------------------|--|
| Description | Obtains the name of the base color space. This is a helper routine for indexed color spaces. |
| Parameters | <i>colorSpace</i> The base color space. |
| Read/Write | Read |
| Return Value | The <i>ASAtom</i> for the name of the base color space. Use ASAtomGetString to obtain a C string for the <i>ASAtom</i> . |
| Exceptions | peErrUnknownPDEColorSpace peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEColorSpaceGetBaseNumComps |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEColorSpaceGetBaseNumComps

```
ASInt32 PDEColorSpaceGetBaseNumComps
( PDEColorSpace colorSpace );
```

| | |
|------------------------|---|
| Description | Gets the number of components in the base color space of an indexed color space. For example, for [/Indexed /DeviceRGB...], the number of components is 3. |
| Parameters | <i>colorSpace</i> The base color space. |
| Read/Write | Read |
| Return Value | Number of components in <i>colorSpace</i> . |
| Exceptions | peErrUnknownPDEColorSpace peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEColorSpaceGetBase PDEColorSpaceGetNumComps |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEColorSpaceGetCosObj

```
void PDEColorSpaceGetCosObj (PDEColorSpace colorSpace,  
    CosObj* cosObjP);
```

Description

Gets a Cos object that describes the color space.

Note: If you get the [PDEColorSpace](#) for an inline image, then get the CosObj for that color space with [PDEColorSpaceGetCosObj](#), this is a limited CosObj. Cos objects that are the result of parsing inline dictionaries in the PDF page contents are a special class of Cos objects. You should never depend on these objects lasting the lifetime of the document. You should extract the information you need from the object immediately and refer to it no further in your code.

Parameters

colorSpace

The color space whose Cos object is obtained.

cosObjP

(Filled by the method) Cos object for the color space.

Read/Write

Read

Return Value

Cos object for *colorSpace*. Any color space that is in the Resources dictionary of the page is returned as a Cos object. Returns the *NULL* Cos object for device color spaces.

Note: This Cos object is subject to the warning above.

Exceptions

[peErrWrongPDEObjectType](#)

Notifications

None

Header File

PERCalls.h

Related Methods

None

Availability

Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDEColorSpaceGetCTable

```
void PDEColorSpaceGetCTable (PDEColorSpace colorSpace,
    ASUns8* colorTableP);
```

| | |
|------------------------|---|
| Description | Obtains the component information for an indexed color space. |
| Parameters | <p><i>colorSpace</i></p> <p>The color space whose component information table is obtained.</p> <p><i>colorTableP</i></p> <p>(Filled by the method) The color lookup table, which is <i>numComps</i> * (<i>hiVal</i> + 1) bytes long, where <i>numComps</i> = number of components in <i>colorSpace</i>.</p> <p>Each entry in the table contains <i>numComps</i> bytes, and the table is indexed 0 to <i>hiVal</i>, where <i>hiVal</i> is the highest index in the color table. The table is indexed from 0 to <i>hival</i>, thus the table contains <i>hival</i> + 1 entries.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrUnknownPDEColorSpace peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | — | — |

PDECOLORSPACEGETHIVAL

```
ASInt32 PDECOLORSPACEGETHIVAL (PDECOLORSPACE colorSpace);
```

| | |
|-----------------|---|
| Description | Obtains the number of entries in the color lookup table for an indexed color space. |
| Parameters | <i>colorSpace</i> An indexed color space. |
| Read/Write | Read |
| Return Value | The number of table entries (<i>hiVal</i>) in the color lookup table. |
| Exceptions | peErrUnknownPDECOLORSPACE |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDECOLORSPACEGETNAME

[ASAtom](#) PDECOLORSPACEGETNAME ([PDECOLORSPACE](#) colorSpace);

| | |
|------------------------|---|
| Description | Obtains the name of a color space object. |
| Parameters | <i>colorSpace</i> A color space object. |
| Read/Write | Read |
| Return Value | The color space object's name. Supports all PDF 1.3 color spaces, which include: Device-dependent names: DeviceCMYK , DeviceGray , DeviceN , DeviceRGB , or ICCBased . Device-independent names: CalGray , CalRGB , or Lab . Special names: Indexed , Pattern , or Separation . Resource names: Arbitrary name specified in page Resources dictionary, including DefaultGray or DefaultRGB . |
| Exceptions | peErrUnknownPDECOLORSPACE |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEColorSpaceGetNumComps

```
ASInt32 PDEColorSpaceGetNumComps ( PDEColorSpace colorSpace );
```

| | |
|------------------------|--|
| Description | Calculates the number of components in a color space. |
| Parameters | <i>colorSpace</i> A color space object. |
| Read/Write | Read |
| Return Value | Number of components in <i>colorSpace</i> . For: DeviceGray, CalGray : Returns 1. DeviceRGB, CalRGB : Returns 3. DeviceCMYK, Lab : Returns 4. DeviceN, ICCBased : Returns the number of components dependent on the specific color space object. Indexed : Returns 1. Call PDEColorSpaceGetBaseNumComps to get the number of components in the base color space. |
| Exceptions | peErrUnknownPDEColorSpace peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEColorSpaceGetBaseNumComps |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEContainer

PDEContainerCreate

```
PDEContainer PDEContainerCreate (ASAtom mcTag, CosObj* cosObjP,  
ASBool isInline);
```

| | |
|------------------------|---|
| Description | Creates a container object. |
| Parameters | <p><i>mcTag</i> Tag name for the container.</p> <p><i>cosObjP</i> Optional Marked Content dictionary for the container.</p> <p><i>isInline</i> If <i>true</i>, emits container into the page content stream inline.</p> |
| Read/Write | Write |
| Return Value | The newly-created container. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEContainerGetContent

[PDEContent](#) PDEContainerGetContent ([PDEContainer](#) pdeContainer);

| | |
|------------------------|---|
| Description | Gets the <i>PDEContent</i> for a <i>PDEContainer</i> . |
| Parameters | <i>pdeContainer</i> The container whose content is obtained. |
| Read/Write | Read |
| Return Value | The <i>PDEContent</i> for the <i>pdeContainer</i> . |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEContainerSetContent |
| Availability | Plug-ins: Available if <i>PL_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEContainerGetDict

```
ASBool PDEContainerGetDict (PDEContainer pdeContainer,
    CosObj* placeDictP, ASBool* isInline);
```

Description Obtains the Marked Content dictionary for a container.

Parameters

pdeContainer
A container.

placeDictP
(Filled by the method) Marked Content dictionary for *pdeContainer*. *NULL* if *pdeContainer* has no Marked Content dictionary.

isInline
(Filled by the method) *true* if the dictionary is inline, *false* otherwise. Undefined if *pdeContainer* has no Marked Content dictionary.

Read/Write Read

Return Value *true* if *pdeContainer* has a Marked Content dictionary, *false* otherwise.

Exceptions [peErrWrongPDEObjectType](#)
[cosErrInvalidObj](#)

Notifications None

Header File *PERCalls.h*

Related Methods [PDEContainerSetDict](#)

Availability Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEContainerGetMCTag

```
ASAtom PDEContainerGetMCTag (PDEContainer pdeContainer);
```

| | |
|------------------------|---|
| Description | Obtains the Marked Content tag for a container. |
| Parameters | <i>pdeContainer</i> A container. |
| Read/Write | Read |
| Return Value | Marked Content tag of <i>pdeContainer</i> . Returns <i>ASAtomNull</i> if <i>pdeContainer</i> has no Marked Content tag. |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEContainerSetMCTag |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEContainerSetContent

```
void PDEContainerSetContent ( PDEContainer pdeContainer ,
    PDEContent pdeContent ) ;
```

| | |
|------------------------|--|
| Description | Sets the content for a container. The existing <i>PDEContent</i> is released by this method. |
| Parameters | <p><i>pdeContainer</i> A container.</p> <p><i>pdeContent</i> The content of <i>pdeContainer</i>.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEContainerGetContent |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDEContainerSetDict

```
void PDEContainerSetDict (PDEContainer pdeContainer,
    CosObj* placeDictP, ASBool isInline);
```

| | |
|------------------------|--|
| Description | Changes the Marked Content dictionary for a container. |
| Parameters | <p><i>pdeContainer</i></p> <p>The container whose dictionary is being changed.</p> <p><i>placeDictP</i></p> <p>Marked Content dictionary being set into <i>pdeContainer</i>.</p> <p><i>isInline</i></p> <p>If <i>true</i>, the dictionary is emitted inline.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEContainerGetDict |
| Availability | Plug-ins: Available if <i>PL_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEContainerSetMCTag

```
void PDEContainerSetMCTag (PDEContainer pdeContainer,
    ASAtom mcTag);
```

| | |
|------------------------|--|
| Description | Sets the Marked Content tag for a <i>PDEContainer</i> . |
| Parameters | <p><i>pdeContainer</i> The container to tag.</p> <p><i>mcTag</i> Marked Content tag.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEContainerGetMCTag |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEContent

PDEContentAddElem

```
void PDEContentAddElem (PDEContent pdeContent,
    ASInt32 addAfterIndex, PDEElement pdeElement);
```

| | |
|------------------------|---|
| Description | Inserts an element into a <i>PDEContent</i> . |
| Parameters | <p><i>pdeContent</i></p> <p>The content to which <i>pdeElement</i> is added.</p> <p><i>addAfterIndex</i></p> <p>Location after which <i>pdeElement</i> is added. <i>addAfterIndex</i> should be <i>kPDEBeforeFirst</i> to add to the beginning of the display list.</p> <p><i>pdeElement</i></p> <p>The element to add to <i>pdeContent</i>. The reference count of <i>pdeElement</i> is incremented.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEContentRemoveElem |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEContentCreate

[PDEContent](#) PDEContentCreate (void);

Description

Creates an empty content object.

Note: Do not use this method to create a PDEContent to be put into a PDPage. Instead, call [PDPageAcquirePDEContent](#).

Call [PDERelease](#) to dispose of the returned content when finished with it.

Parameters

None

Read/Write

Write

Return Value

An empty content object.

Exceptions

[peErrPStackUnderflow](#)

Notifications

None

Header File

PEWCalls.h

Related Methods

[PDEContentCreateFromCosObj](#)
[PDPageAcquirePDEContent](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEContentCreateFromCosObj

```
PDEContent PDEContentCreateFromCosObj (const CosObj* contents,
const CosObj* resources);
```

| | |
|------------------------|--|
| Description | Creates a content object from a Cos object. This is the main method for obtaining a <i>PDEContent</i> . Call PDERelease with the <i>PDEContent</i> when finished with it. |
| Parameters | <p><i>contents</i></p> <p>Cos object that is the source for the content. May be a page contents, a Form <i>XObject</i>, or a Type 3 font CharProc.</p> <p><i>resources</i></p> <p>The object's Resources dictionary. If the Form or Type 3 font contains a Resources dictionary, this dictionary must be passed in <i>resources</i>. Otherwise, it must be the page resources object of the page containing the <i>contents</i>.</p> |
| Read/Write | Read |
| Return Value | The content from the Cos object. |
| Exceptions | peErrPStackUnderflow |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEContentCreate PDEContentToCosObj PDEEnumElements |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | — | — |

PDEContentGetAttrs

```
void PDEContentGetAttrs (PDEContent pdeContent,
    PDEContentAttrsP attrsP, ASUns32 attrsSize);
```

| | |
|------------------------|--|
| Description | Obtains the attributes of a content. |
| Parameters | <p><i>pdeContent</i></p> <p>A content object.</p> <p><i>attrsP</i></p> <p>(Filled by the method) Pointer to a PDEContentAttrs structure containing the attributes of the content.</p> <p><i>attrsSize</i></p> <p>Size of the <i>attrsP</i> buffer, in bytes.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEContentGetDefaultColorSpace

[PDEColorSpace](#) PDEContentGetDefaultColorSpace
 ([PDEContent](#) pdeContent, [ASAtom](#) colorSpaceName);

- Description** Gets a default colorspace from a *PDEContent*.
 See Section 7.11 in the [Portable Document Format Reference Manual](#) for more information about default color spaces.
- Parameters**
- pdeContent*
 A content object.
- colorSpaceName*
 An *ASAtom* for the name of the desired colorspace. Must be an *ASAtom* for one of **DefaultRGB**, **DefaultCMYK**, or **DefaultGray**.
- Read/Write** Read
- Return Value** The desired colorspace in *pdeContent*. Returns *NULL* if *colorSpaceName* does not correspond to a known default, such as **DefaultRGB**.
- Exceptions** None
- Notifications** None
- Header File** *PERCalls.h*
- Related Methods** [PDEContentGetNumElems](#)
- Availability** Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEContentGetElem

```
PDEElement PDEContentGetElem (PDEContent pdeContent,
    ASInt32 index);
```

Description

Gets the requested element from a content.

Note: This method does not change the reference count of the element.

Note: This method does not copy the element. The only thing that can be done with elements obtained in this manner is to get and set their attributes, matrices, and states.

Parameters

pdeContent

A content object.

index

Index of element to obtain.

Read/Write

Read

Return Value

The requested element.

Exceptions

[peErrWrongPDEObjectType](#)

Notifications

None

Header File

PERCalls.h

Related Methods

[PDEContentGetNumElems](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEContentGetNumElems

```
ASInt32 PDEContentGetNumElems ( PDEContent pdeContent );
```

| | |
|------------------------|---|
| Description | Gets the number of elements in a <i>PDEContent</i> . |
| Parameters | <i>pdeContent</i> A content object. |
| Read/Write | Read |
| Return Value | Number of elements in <i>pdeContent</i> . |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEContentGetElem |
| Availability | Plug-ins: Available if <i>PL_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEContentGetResources

```
ASInt32 PDEContentGetResources (PDEContent pdeContent,
    ASInt32 type, PDEObject* resourcesP);
```

| | |
|------------------------|---|
| Description | Obtains the number of resources of the specified type and, optionally, pointers to the resource objects. |
| Parameters | <p><i>pdeContent</i></p> <p>A content object.</p> <p><i>type</i></p> <p>Type of resources to query or obtain: PDEFont, PDEXObject, or PDEColorSpace. Must be one of PDEContentGetResourceFlags.</p> <p><i>resourcesP</i></p> <p>(Filled by the method) If non-NULL, it must point to an array of <i>PDEObject</i> pointers. On return, the array contains pointers to the requested resources. If <i>resourcesP</i> is NULL, only the number of resources of <i>type</i> is returned.</p> |
| Read/Write | Read |
| Return Value | Number of resources of <i>type</i> returned in <i>resourcesP</i> . |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEContentRemoveElem

```
void PDEContentRemoveElem (PDEContent pdeContent,  
    ASInt32 index);
```

Description

Removes an element from a *PDEContent*.

Note: This decrements the reference count of the element removed.

Parameters

pdeContent

A content object.

index

Index in *pdeContent* of the element to remove, whose reference count is decremented.

Read/Write

Write

Return Value

None

Exceptions

[peErrWrongPDEObjectType](#)

Notifications

None

Header File

PEWCalls.h

Related Methods

[PDEContentAddElem](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDEContentToCosObj

```
void PDEContentToCosObj (PDEContent pdeContent, ASUns32 flags,
PDEContentAttrsP attrs, ASUns32 attrsSize, CosDoc cosDoc,
PDEFilterArrayP filtersP, CosObj* contentsP,
CosObj* resourcesP);
```

Description

This is the main method for converting a *PDEContent* into PDF contents and resources.

Note: Do not use this method to put a PDEContent into a PDPage. Instead, call [PDPageSetPDEContent](#).

This method does not change the *PDEContent* object.

The caller of this function is responsible for adding the contents and the resources returned from this method to the Page Object.

Parameters

pdeContent

A content object.

flags

Flags specifying the type of object to create (page contents, form, or charproc) and how it is created. Must be one or more of [PDEContentToCosObjFlags](#).

You should always pass in the *kPDEContentDoNotResolveForms* flag unless you need to generate PDF 1.1 compatible files (Acrobat 2.x). If you do not set this flag, then this method must look at all pages in the document to see if any forms are shared on any pages. If so, it must generate consistent names for use in shared forms.

Use the *kPDEContentDoNotResolveForms* flag when you call *PDEContentToCosObj* to get a Cos object from a *PDEContent*—not when you get a Cos object for an XObject Form itself.

attrs

A pointer to a [PDEContentAttrs](#) structure that contains the appropriate form attributes or cache device/charwidth attributes, and so on. If zero, no attributes are set.

attrsSize

Size of the *attrs* buffer, in bytes. Zero if *attrs* is zero.

cosDoc

Document in which the contents and resources are created.

filtersP

A pointer to a [PDEFilterArray](#) structure that specifies which filters to use in encoding the contents; may be *NULL*. If *filtersP* contains any *encodeParms*, they must belong to *cosDoc*.

contentsP

(Filled by the method) Cos object for the resulting contents in *pdeContent*.

resourcesP

(Filled by the method) Cos object for the resulting resources in *pdeContent*.

Read/Write

Write

Return Value

None

Exceptions

[peErrUnknownResType](#)
[pageErrErrorParsingImage](#)
[pdErrBadResMetrics](#)
[peErrWrongPDEObjectType](#)
[peErrUnknownPDEColorSpace](#)

Notifications

None

Header File

PEWCalls.h

Related Methods

[PDEContentCreateFromCosObj](#)

Example

```
PDEContentToCosObj( pdeContent, 0, NULL, 0,
    0, dP/*CosDoc*/, 0, &contents, &resources
);
```

```
/*Get the CosObj for the page */
cosPage = PDPageGetCosObj (pdPage);
CosDictPut(cosPage,
    ASAtomFromString("Contents"), contents);
```

```
CosDictPut(cosPage,
    ASAtomFromString("Resources"),
    resources);
```

Availability

Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEDeviceNColors

PDEDeviceNColorsCreate

```
PDEDeviceNColors PDEDeviceNColorsCreate (ASFixed* pColorValues,  
ASInt32 numValues);
```

| | |
|------------------------|--|
| Description | Creates an object that can be used to store <i>n</i> color components when in a <i>PDEDeviceNColors</i> color space. |
| Parameters | <i>pColorValues</i> Pointer to an array of <i>ASFixed</i> values. <i>numValues</i> The length of the array. |
| Return Value | An object containing values specifying a color in a <i>PDEDeviceNColors</i> color space. |
| Exceptions | genErrNoMemory |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEDeviceNColorsGetColorValue |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEDeviceNColorsGetColorValue

[ASFixed](#) PDEDeviceNColorsGetColorValue ([PDEDeviceNColors](#) colors,
ASInt32 index);

| | |
|------------------------|---|
| Description | Gets the value of a color component of a <i>PDEDeviceNColors</i> color space. |
| Parameters | <i>colors</i> A <i>PDEDeviceNColors</i> object returned by PDEDeviceNColorsCreate . <i>index</i> Index of the color component to return. |
| Return Value | The value of the requested color component. |
| Exceptions | genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEDeviceNColorsCreate |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEElement

PDEElementCopy

```
PDEElement PDEElementCopy (PDEElement pdeElement,  
ASUns32 flags);
```

| | |
|------------------------|---|
| Description | Makes a copy of an element. The caller is responsible for releasing the copy with PDERelease . |
| Parameters | <p><i>pdeElement</i> The element to copy.</p> <p><i>flags</i> Bit field of PDEElementCopyFlags.</p> |
| Read/Write | Write |
| Return Value | A copy of <i>pdeElement</i> . |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEElementGetBBox

```
void PDEElementGetBBox (PDEElement pdeElement,
    ASFixedRectP bboxP);
```

| | |
|------------------------|---|
| Description | Obtains the bounding box for an element. The returned bounding box is guaranteed to encompass the element, but is <i>not</i> guaranteed to be the smallest box that could contain the element. For example, for an arc, <i>bboxP</i> encloses the bezier control points—not just the curve itself. |
| Parameters | <p><i>pdeElement</i></p> <p>An element whose bounding box is obtained.</p> <p><i>bboxP</i></p> <p>(Filled by the method) Pointer to a ASFixedRect structure specifying the bounding box of <i>pdeElement</i>, specified in user space coordinates.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEElementGetClip

[PDEClip](#) PDEElementGetClip ([PDEElement](#) pdeElement);

Description

Gets the current clip for an element.

Note: This method does not change the reference count of the clip object.

Parameters

pdeElement

An element whose clip is obtained.

Read/Write

Read

Return Value

Clip object for *pdeElement*.

Exceptions

[peErrWrongPDEObjectType](#)

Notifications

None

Header File

PERCalls.h

Related Methods

[PDEElementIsAtRect](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEElementGetGState

```
void PDEElementGetGState (PDEElement pdeElement,  
    PDEGraphicStateP stateP, ASUns32 stateSize);
```

Description Obtains the graphics state information for an element. This method is not valid for a [PDEText](#) or a [PDEImage](#) element.

Parameters

pdeElement
An element whose graphics state is obtained.

stateP
(Filled by the method) Pointer to a [PDEGraphicState](#) structure that contains graphics state information for *pdeElement*.
This [PDEGraphicState](#) may contain PDEObjects for color spaces or an *ExtGState*. They are *not* acquired by this method. You *must* acquire them before making any call that could result in their being released—such as releasing *pdeElement*.

stateSize
Size of the *stateP* buffer, in bytes.

Read/Write Read

Return Value None

Exceptions [peErrWrongPDEObjectType](#)
[genErrBadParm](#)

Notifications None

Header File *PERCalls.h*

Related Methods [PDEElementSetGState](#)

Availability Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDEElementGetMatrix

```
void PDEElementGetMatrix (PDEElement pdeElement,
    ASFixedMatrixP matrixP);
```

| | |
|------------------------|---|
| Description | Obtains the transformation matrix for an element. This matrix provides the transformation from page space to user space for the element. If there is no cm operator (concatmatrix) in the page stream, the matrix is the identity matrix. This method is not valid for a PDEText element. |
| Parameters | <p><i>pdeElement</i> An element whose transformation matrix is obtained.</p> <p><i>matrixP</i> (Filled by the method) Pointer to ASFixedMatrix that holds a transformation matrix for <i>pdeElement</i>. If <i>pdeElement</i> is a text object, returns the identity matrix.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEElementSetMatrix |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEElementIsAtPoint

```
ASBool PDEElementIsAtPoint (PDEElement elem, ASFixedPoint point);
```

| | |
|------------------------|--|
| Description | Tests whether a point is on an element. |
| Parameters | <p><i>elem</i></p> <p>The element to test.</p> <p>If <i>PDEElement</i> is a <i>PDEText</i> or <i>PDEImage</i>, it uses the bounding box of the <i>PDEElement</i> to make the check. If the <i>PDEElement</i> is a <i>PDEPath</i> and it is stroked, it checks if the point is on the path. If the <i>PDEElement</i> is a <i>PDEPath</i> and it is filled, it checks if the point is in the fill area, taking into consideration whether it is filled using the non-zero winding number rule or the even-odd rule.</p> <p><i>point</i></p> <p>The point, specified in user space coordinates.</p> |
| Return Value | <i>true</i> if the point is on the element, <i>false</i> otherwise. |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEElementIsAtRect PDETextIsAtPoint PDETextIsAtRect |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEElementIsAtRect

```
ASBool PDEElementIsAtRect (PDEElement elem, ASFixedRect rect);
```

Description Tests whether any part of a rectangle is on an element.

Parameters *elem*

The element to test.

If *PDEElement* is a *PDEText* or *PDEImage*, it uses the bounding box of the *PDEElement* to make the check. If the *PDEElement* is a *PDEPath* and it is stroked, it checks if the point is on the path. If the *PDEElement* is a *PDEPath* and it is filled, it checks if the point is in the fill area, taking into consideration whether it is filled using the non-zero winding number rule or the even-odd rule.

rect

The rectangle, specified in user space coordinates.

Return Value *true* if any part of the rectangle is on the element, *false* otherwise.

Exceptions [peErrWrongPDEObjectType](#)

Notifications None

Header File *PERCalls.h*

Related Methods [PDEElementIsAtPoint](#)
[PDETextIsAtPoint](#)
[PDETextIsAtRect](#)

Availability Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEElementSetClip

```
void PDEElementSetClip (PDEElement pdeElement,  
    PDEClip pdeClip);
```

| | |
|------------------------|---|
| Description | Sets the current clip for an element. <i>pdeElement</i> 's previous clip's reference count is decremented (if it had one), and <i>pdeClip</i> 's reference count is incremented. |
| Parameters | <i>pdeElement</i> An element whose clip is set. <i>pdeClip</i> The clip to set for <i>pdeElement</i> . |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEElementGetClip |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEElementSetGState

```
void PDEElementSetGState (PDEElement pdeElement,
    PDEGraphicStateP stateP, ASUns32 stateSize);
```

Description

Sets the graphics state information for an element.

Note: This method causes any of stateP's color space or ExtGState objects to have their reference count incremented.

This method is not valid for a [PDEText](#) element.

Parameters

pdeElement

An element whose graphics state is set.

stateP

Pointer to a [PDEGraphicState](#) structure with graphics state information to set for *pdeElement*. Any of *stateP*'s color space or *ExtGState* objects have their reference count incremented.

stateSize

Size of the *stateP* buffer, in bytes.

Read/Write

Write

Return Value

None

Exceptions

[peErrWrongPDEObjectType](#)
[genErrBadParm](#)

Notifications

None

Header File

PEWCalls.h

Related Methods

[PDEElementGetGState](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | — | — |

PDEElementSetMatrix

```
void PDEElementSetMatrix (PDEElement pdeElement,
    ASFixedMatrixP matrixP);
```

| | |
|------------------------|---|
| Description | Sets the transformation matrix for an element. The element may not be a PDEContainer , a PDEText , or a PDEPlace . |
| Parameters | <p><i>pdeElement</i> An element whose transformation matrix is set.</p> <p><i>matrixP</i> Pointer to ASFixedMatrix that holds the transformation matrix to set for <i>pdeElement</i>.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEElementGetMatrix |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEExtGState

PDEExtGStateCreate

[PDEExtGState](#) PDEExtGStateCreate ([CosObj](#)* cosObjP);

| | |
|------------------------|---|
| Description | Creates a new <i>PDEExtGState</i> from a Cos object. See Section 7.14 in the Portable Document Format Reference Manual for more information about extended graphics states. |
| Parameters | <i>cosObjP</i> A Cos object for a PDEElement . |
| Read/Write | Write |
| Return Value | The <i>PDEExtGState</i> for <i>cosObjP</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEExtGStateGetCosObj |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEExtGStateGetCosObj

```
void PDEExtGStateGetCosObj (PDEExtGState extGState,  
    CosObj* cosObjP);
```

| | |
|------------------------|---|
| Description | Obtains a Cos object for a <i>PDEExtGState</i> . |
| Parameters | <i>extGState</i> A <i>PDEExtGState</i> whose Cos object is obtained. <i>cosObjP</i> (Filled by the method) Cos object for <i>extGState</i> . |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEExtGStateCreate |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEFont

PDFontCreate

```
PDFont PDFontCreate (PDFontAttrsP attrsP, ASUns32 attrsSize,
    ASInt32 firstChar, ASInt32 lastChar, ASInt16* widthsP,
    char** encoding, ASAtom encodingBaseName, ASStm fontStm,
    ASInt32 len1, ASInt32 len2, ASInt32 len3);
```

Description

Creates a new *PDFont* from the specified parameters.

The *PDFont* may be represented as an embedded font (a **FontFile** entry in the font descriptor of the PDF file). To create a *PDFont* that is stored as an embedded font, the **FontFile** stream may be passed in *fontStm*, and the *len1*, *len2*, and *len3* parameters contain the **Length1**, **Length2**, and **Length3** values of the **FontFile** stream attributes dictionary. See Section 7.10 in the [Portable Document Format Reference Manual](#) for more information about embedded fonts.

The caller must close *fontStm* with [ASStmClose](#) after invoking *PDFontCreate*.

Parameters

attrsP

Pointer to a [PDFontAttrs](#) structure for the font attributes.

attrsSize

Size of the *attrsP* buffer, in bytes.

firstChar

First character index for the widths array, *widthsP*.

lastChar

Last character index for the widths array, *widthsP*.

widthsP

Pointer to widths array.

encoding

An array of 256 pointers to glyph names specifying the custom encoding. If any pointer is *NULL*, no encoding information is written for that entry.

encodingBaseName

Encoding base name if the encoding is a custom encoding. If *encoding* is *NULL*, *encodingBaseName* is used as the value of the encoding, and must be one of "WinAnsiEncoding," "MacRomanEncoding," or "MacExpertEncoding." If no Encoding value is desired, use *ASAtomNull*.

fontStm

Stream with font information.

len1

Length in bytes of the ASCII portion of the Type 1 font file after it has been decoded. For other font formats, such as TrueType or CFF, only *len1* is used, and it is the size of the font.

len2

Length in bytes of the encrypted portion of the Type 1 font file after it has been decoded.

len3

Length in bytes of the portion of the Type 1 font file that contains the 512 zeros, plus the **cleartomark** operator, plus any following data.

Read/Write

Write

Return Value

The specified *PDEFont*.

Exceptions

[peErrCantCreateFontSubset](#)
[peErrCantGetAttrs](#)
[peErrCantGetWidths](#)
[peErrCantEmbedFont](#)
[genErrResourceLoadFailed](#)

Notifications

None

Header File

PEWCalls.h

Related Methods

[PDEFontCreateFromCosObj](#)
[PDEFontCreateFromSysFont](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDFontCreateFromCosObj

```
PDFont PDFontCreateFromCosObj (const CosObj* cosObjP);
```

| | |
|------------------------|---|
| Description | Creates a <i>PDFont</i> corresponding to a Cos object of type Font. |
| Parameters | <i>cosObjP</i> Cos object for which a <i>PDFont</i> is created. |
| Read/Write | Write |
| Return Value | <i>PDFont</i> created from <i>cosObjP</i> . |
| Exceptions | peErrCantCreateFontSubset peErrCantGetAttrs peErrCantGetWidths peErrCantEmbedFont genErrBadParm genErrResourceLoadFailed |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDFontCreate PDFontCreateFromSysFont PDFontGetCosObj |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDFontCreateFromSysFont

```
PDFont PDFontCreateFromSysFont (PDSysFont sysFont ,
ASUns32 flags);
```

Description

Gets a *PDFont* corresponding to a font in the system.

The [PDFontCreateFlags](#) flags *kPDFontCreateEmbedded* and *kPDFontWillSubset* must *both* be set in order to subset a font.

If you create a *PDFont* that is subsetting, call [PDFontSubsetNow](#) on this font afterwards.

Note: If you have environment with no Acrobat Language kit installed, trying to acquire a PDFont from the system font may raise an exception for some of the OS fonts. In other words, if you use [PDEnumSysFonts](#) to get the system font attributes, not all of the system fonts will necessarily be used to create the PDFont.

Parameters

sysFont

A *PDSysFont* object referencing a system font.

flags

Indicates whether to embed the font and whether to subset the font. Must be one of [PDFontCreateFlags](#). If you want to subset a font, set *both* the *kPDFontCreateEmbedded* and *kPDFontWillSubset* flags.

Read/Write

Write

Return Value

The *PDFont* corresponding to *sysFont*.

Exceptions

[peErrCantCreateFontSubset](#)
[peErrCantGetAttrs](#)
[peErrCantGetWidths](#)
[peErrCantEmbedFont](#)
[genErrBadParm](#)
[genErrResourceLoadFailed](#)

Notifications

None

Header File

PEWCalls.h

Related Methods

[PDFontCreateFromCosObj](#)
[PDEnumSysFonts](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDFontCreateFromSysFontEx

```
PDFont PDFontCreateFromSysFontEx (PDSysFont sysFont,
    ASUns32 flags, ASAtom snapshotName, ASFixed* mmDesignVec);
```

Description

Creates a *PDFont* corresponding to a font in the system.

If the font is a multiple master font, *mmDesignVector* points to the design vector, whose length must equal the number of design axes of the font.

The PDFontCreateFlags flags *kPDFontCreateEmbedded* and *kPDFontWillSubset* must *both* be set in order to subset a font.

If you create a *PDFont* that is subsetting, call PDFontSubsetNow on this font afterwards.

Note: If you have environment with no Acrobat Language kit installed, trying to acquire a PDFont from the system font may raise an exception for some of the system fonts. In other words, if you use PDEnumSysFonts to get the system font attributes, not all of the system fonts are necessarily used to create the PDFont.

Parameters

sysFont

A *PDSysFont* object referencing a system font.

flags

Indicates whether to embed the font and whether to subset the font. Must be one of PDFontCreateFlags. If you want to subset a font, set *both* the *kPDFontCreateEmbedded* and *kPDFontWillSubset* flags.

snapshotName

Name to be associated with this particular instantiation of the *PDFont*.

mmDesignVec

Pointer to multiple master font design vector.

Read/Write

Write

Return Value

The *PDFont* corresponding to *sysFont*.

Exceptions

[peErrCantCreateFontSubset](#)
[peErrCantGetAttrs](#)
[peErrCantGetWidths](#)
[peErrCantEmbedFont](#)
[genErrBadParm](#)
[genErrResourceLoadFailed](#)

Notifications

None

Header File

PEWCalls.h

Related Methods

[PDEFontCreateFromCosObj](#)
[PDEnumSysFonts](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDFontCreateWithParams

[PDFont](#) PDFontCreateWithParams ([PDFontCreateParams](#) params);

Description

Creates a new *PDFont* from *params*.

The *PDFont* may be represented as an embedded font (a **FontFile** value in PDF). To create a *PDFont* that will be stored as an embedded font, the **FontFile** stream may be passed as *fontStm*, and the *len1*, *len2*, and *len3* parameters contain the **Length1**, **Length2**, and **Length3** values of the **FontFile**. The caller must close the *fontStm* after calling this method. This method supports multibyte fonts.

This method extends [PDFontCreate](#) to support multibyte fonts.

Parameters

params

Pointer to a structure containing all font parameters necessary to fully define a font.

Return Value

A *PDFont* object of the font described by the parameters.

Exceptions

[peErrCantCreateFontSubset](#)
[peErrCantGetAttr](#)
[peErrCantGetWidths](#)
[peErrCantEmbedFont](#)
[genErrBadParm](#)
[genErrResourceLoadFailed](#)

Notifications

None

Header File

PEWCalls.h

Related Methods

[PDFontCreate](#)
[PDFontCreateFromCosObj](#)
[PDFontCreateFromSysFont](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|---------|
| Adobe PDF Library | — | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDFontGetAttrs

```
void PDFontGetAttrs (PDFont font, PDFontAttrsP attrsP,
    ASUns32 attrsSize);
```

| | |
|------------------------|---|
| Description | Obtains the attributes for a font object. |
| Parameters | <p><i>font</i></p> <p>A <i>PDFont</i> whose attributes are found.</p> <p><i>attrsP</i></p> <p>(Filled by the method) Pointer to a PDFontAttrs structure for the font attributes.</p> <p><i>attrsSize</i></p> <p>Size of the <i>attrsP</i> buffer, in bytes.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrCantGetAttrs genErrBadParm genErrResourceLoadFailed |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDFontGetNumCodeBytes |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDFontGetCosObj

```
void PDFontGetCosObj (PDFont font, CosObj* cosObjP);
```

| | |
|------------------------|--|
| Description | Obtains a Cos object for a <i>PDFont</i> . |
| Parameters | <p><i>font</i></p> <p>A <i>PDFont</i> whose Cos object is obtained.</p> <p><i>cosObjP</i></p> <p>(Filled by the method) The Cos object corresponding to <i>font</i>.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | genErrResourceLoadFailed peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDFontCreateFromCosObj |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDFontGetNumCodeBytes

```
ASInt16 PDFontGetNumCodeBytes (PDFont font, ASUns8* text,
    ASInt32 len);
```

Description Gets the number of bytes comprising the next code in a string of single or mulitbyte character codes.

Parameters *font*

A *PDFont* object returned from one of the [PDFontCreate](#) methods.

text

Pointer into a string of characters.

len

The length, in bytes, of the string of characters, starting with the character pointed to by *text*.

Return Value Number of bytes in the next character code pointed to by *text*.

Exceptions [genErrNoMemory](#)

Notifications None

Header File *PERCalls.h*

Related Methods [PDFontIsMultiByte](#)
[PDFontSumWidths](#)
[PDFontGetNumCodeBytes](#)

Availability Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDFontGetOneByteEncoding

```
ASBool PDFontGetOneByteEncoding (PDFont font,  
    ASAtom* encodingDelta);
```

Description Gets an array of delta encodings for the given one byte *PDFont*.

Parameters *font*

A *PDFont* object returned from one of the [PDFontCreate](#) methods.

encodingDelta

(Filled by the method) Pointer to an *ASAtom* array that is filled with the delta encodings for *font*. Each entry is the *ASAtom* for a glyph name that differs from the base encoding. See Section 7.8 in the [Portable Document Format Reference Manual](#) for more information about font encodings.

The array should be allocated to hold 256 entries.

Return Value *true* if *encodingDelta* is filled, *false* otherwise.

Exceptions [genErrNoMemory](#)

Notifications None

Header File *PERCalls.h*

Related Methods [PDFontIsMultiByte](#)
[PDFontSumWidths](#)
[PDFontGetNumCodeBytes](#)

Availability Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDFontGetWidths

```
void PDFontGetWidths (PDFont font, ASInt16* widthsP);
```

| | |
|------------------------|--|
| Description | Gets the widths for a font object. |
| Parameters | <p><i>font</i></p> <p>A <i>PDFont</i> whose widths are found.</p> <p><i>widthsP</i></p> <p>(Filled by the method) Pointer to widths array. <i>widthsP</i> must have room for 256 values. The widths are returned in character space and 1000 EM units. An EM is a typographic unit of measurement equal to the size of a font. To convert to text space, divide the value returned by 1000. To convert to user space, multiply the value by the font size.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrCantGetWidths genErrBadParm genErrResourceLoadFailed |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDFontCreateWithParams |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDFontIsMultiByte

```
ASBool PDFontIsMultiByte (PDFont font);
```

| | |
|------------------------|---|
| Description | Tests whether a font contains any multibyte characters. |
| Parameters | <p><i>font</i></p> <p>A <i>PDFont</i> object returned from one of the PDFontCreate methods to test.</p> |
| Return Value | <i>true</i> if the font contains any multibyte characters, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDFontGetNumCodeBytes PDFontSumWidths |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDFontSubsetNow

```
void PDFontSubsetNow (PDFont font, CosDoc cosDoc);
```

| | |
|------------------------|--|
| Description | <p>Subsets a given <i>PDFont</i> in a <i>CosDoc</i>.</p> <p>If you created <i>font</i> with PDFontCreateFromSysFont, you must have set <i>both</i> the <i>kPDFontCreateEmbedded</i> and <i>kPDFontWillSubset</i> set in the <i>flags</i> parameter to be able to subset <i>font</i>.</p> |
| Parameters | <p><i>font</i></p> <p>The <i>PDFont</i> to subset.</p> <p><i>cosDoc</i></p> <p>The <i>CosDoc</i> whose font is subsetted.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrCantCreateFontSubset peErrCantGetAttrs peErrCantGetWidths peErrCantEmbedFont genErrBadParm genErrResourceLoadFailed |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDFontCreateFromSysFont |

Example

```
PDDoc pdoc;
PDPage pdPage;
PDEFont gFont;

PDPageAcquirePDEContent(pdPage,
    extensionID);

/* Make sure you specify both font flags */
gFont = PDEFontCreateFromSysFont(sysFont,
    (kPDEFontCreateEmbedded|
    kPDEFontWillSubset));

/* Add your text objects here */
...

PDPageSetPDEContent(pdPage, extensionID);
/* Make sure to call PDEFontSubsetNow after
    setting the PDEContent but before
    releasing it */
PDEFontSubsetNow(gFont,
    PDDocGetCosDoc(pdDoc));

PDERelease(gFont);

PDPageReleasePDEContent(pdPage,
    extensionID);
```

Availability

Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDFontSumWidths

```
ASInt32 PDFontSumWidths (PDFont font, ASUns8* text,
    ASInt32 len);
```

| | |
|------------------------|---|
| Description | Gets the sum to the widths of <i>len</i> characters from a string of single or mulitbyte characters. |
| Parameters | <p><i>font</i></p> <p>A <i>PDFont</i> object returned from one of the PDFontCreate methods.</p> <p><i>text</i></p> <p>Pointer into a string of characters.</p> <p><i>len</i></p> <p>Number of characters in the string.</p> |
| Return Value | Width of text string in EM space (the width of “M” is about 1000 EM units). |
| Exceptions | genErrNoMemory pdErrBadResMetrics genErrResourceLoadFailed peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDFontGetNumCodeBytes PDFontIsMultiByte |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDFindSysFontForPDEFont

```
PDSysFont PDFindSysFontForPDEFont (PDEFont font,  
ASUns32 flags);
```

| | |
|------------------------|---|
| Description | Find a system font that matches the requested <i>PDEFont</i> . The method gets the <i>PDSysFont</i> rather than acquires it, so do <i>not</i> call PDERelease on the returned <i>PDSysFont</i> when done with it. |
| Parameters | <i>font</i> A <i>PDEFont</i> whose matching system font is found. <i>flags</i> Bit field comprised of PDSysFontMatchFlags values. <ul style="list-style-type: none">• kPDSysFontMatchNameAndCharSet• kPDSysFontMatchFontType /* Type 1 */• PDSysFontMatchFlags Passing zero will match font by name only. |
| Read/Write | Write |
| Return Value | The system font corresponding to <i>font</i> . |
| Exceptions | peErrCantGetAttrs genErrBadParm genErrResourceLoadFailed |
| Notifications | None |
| Header File | <i>PSFCalls.h</i> |
| Related Methods | PDFindSysFont |
| Availability | Plug-ins: Available if <i>PI_PDSYSFONT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | — | — |

PDEForm

PDEFormCreateFromCosObj

```
PDEForm PDEFormCreateFromCosObj (const CosObj* xObjectP,  
const CosObj* resourcesP, ASFixedMatrixP matrixP);
```

| | |
|------------------------|--|
| Description | Creates a new form from an existing Cos object. |
| Parameters | <p><i>xObjectP</i> Cos object from which a <i>PDEForm</i> is created.</p> <p><i>resourcesP</i> The <i>xObjectP</i>'s Resources dictionary.</p> <p><i>matrixP</i> Pointer to ASFixedMatrix that holds the transformation matrix to use for the form.</p> |
| Read/Write | Write |
| Return Value | The newly-created form object. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEFormGetCosObj |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEFormGetContent

[PDEContent](#) PDEFormGetContent ([PDEForm](#) form);

| | |
|------------------------|--|
| Description | Obtains a <i>PDEContent</i> object for a form. |
| Parameters | <i>form</i> The form whose content is obtained. |
| Read/Write | Write |
| Return Value | Content for <i>form</i> . |
| Exceptions | peErrWrongPDEObjectType peErrPStackUnderflow |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEFormGetCosObj

```
void PDEFormGetCosObj (PDEForm form, CosObj* cosObjP);
```

| | |
|------------------------|---|
| Description | Obtains a Cos object for a form. |
| Parameters | <p><i>form</i></p> <p>The form whose Cos object is obtained.</p> <p><i>cosObjP</i></p> <p>(Filled by the method) Cos object for the form.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEFormCreateFromCosObj |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEGroup

PDEGroupCreate

[PDEGroup](#) PDEGroupCreate (void);

| | |
|------------------------|--|
| Description | Creates a <i>PDEGroup</i> object. |
| Parameters | None |
| Return Value | The newly-created <i>PDEGroup</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEGroupSetContent |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEGroupGetContent

[PDEContent](#) PDEGroupGetContent ([PDEGroup](#) pdeGroup);

| | |
|------------------------|---|
| Description | Gets the <i>PDEContent</i> for a <i>PDEGroup</i> . |
| Parameters | <i>pdeGroup</i> The group whose content is obtained. |
| Return Value | The <i>PDEContent</i> in <i>pdeGroup</i> . |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEGroupSetContent |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEGroupSetContent

```
void PDEGroupSetContent ( PDEGroup pdeGroup,  
    PDEContent pdeContent );
```

| | |
|------------------------|--|
| Description | Sets the <i>PDEContent</i> for a <i>PDEGroup</i> . The existing <i>PDEContent</i> is released by this method. |
| Parameters | <i>pdeContainer</i> A container object. <i>pdeContent</i> The content to set for <i>pdeGroup</i> . |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEGroupGetContent |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEImage

PDEImageCreate

```
PDEImage PDEImageCreate (PDEImageAttrsP attrsP,  
ASUns32 attrsSize, ASFixedMatrixP matrixP, ASUns32 flags,  
PDEColorSpace colorSpace, PDEColorValueP colorValueP,  
PDEFilterArrayP filtersP, ASStm dataStm, ASUns8* data,  
ASUns32 encodedLen);
```

Description

Creates an image object.

The image data may be specified as a stream or as a buffer. If *dataStm* is non-*NULL*, *data* is ignored.

See [PDEImageSetDataStm](#) for information on handling the stream.

The caller must dispose of *datStm* after calling this method.

Parameters

attrsP

Pointer to [PDEImageAttrs](#) with attributes of the image.

attrsSize

Size of the *attrsP* buffer, in bytes.

matrixP

Pointer to [ASFixedMatrix](#) that holds the transformation matrix to use for the image.

flags

[PDEImageDataFlags](#) flags. If the *kPDEImageEncodedData* flag is set, and the data is provided directly (not as a stream), then *encodedLen* must specify the length of *data*.

colorSpace

Color space of the image. When the image is an imagemask, *colorSpace* is the color space of the *colorValueP* argument.

colorValueP

Pointer to [PDEColorValue](#) structure. If the image is an imagemask, *colorValueP* must be provided.

filtersP

Pointer to [PDEFilterArray](#) structure that specifies which filters to use in encoding the contents; may be *NULL*. Filters will be used to encode the data in the order in which they are specified in the array.

dataStm

Stream holding the image data.

data

Image data. If *dataStm* is non-*NULL*, *data* is ignored.

encodedLen

Encoded length of *data*, in bytes.

Read/Write

Write

Return Value

The image.

Exceptions

[peErrUnknownPDEColorSpace](#)
[pageErrReadLessImageData](#)
[peErrWrongPDEObjectType](#)
[genErrBadParm](#)

Notifications

None

Header File

PEWCalls.h

Related Methods

[PDEImageCreateFromCosObj](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEImageCreateFromCosObj

```
PDEImage PDEImageCreateFromCosObj (const CosObj* imageObjP,  
    ASFixedMatrixP matrixP, PDEColorSpace colorSpace,  
    PDEColorValueP colorValueP);
```

| | |
|------------------------|---|
| Description | Creates an image object from a Cos object. |
| Parameters | <p><i>imageObjP</i> Cos object for the image.</p> <p><i>matrixP</i> Pointer to <u>ASFixedMatrix</u> that holds the transformation matrix to use for the image.</p> <p><i>colorSpace</i> Color space used for the image.</p> <p><i>colorValueP</i> Pointer to <u>PDEColorValue</u> structure. If the image is an imagemask, <i>colorValueP</i> must be provided.</p> |
| Read/Write | Write |
| Return Value | Image corresponding to the Cos object. |
| Exceptions | peErrUnknownPDEColorSpace pageErrReadLessImageData peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEImageCreate PDEImageGetCosObj |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDEImageDataIsEncoded

```
ASBool PDEImageDataIsEncoded ( PDEImage image,
                               ASUns32* encodedLenP );
```

Description

Determines if image data is encoded or not. Used only for inline images; not relevant to *XObject* images.

Always returns *false* for *XObject* images; *XObject* image data can be obtained from [PDEImageGetData](#) or [PDEImageGetDataStm](#), either encoded or decoded.

If an inline image is obtained via the [PDEContentCreateFromCosObj](#) or related methods, the inline image data is always decoded. That is, if PDFEdit parses the stream, the data is always decoded. Only if [PDEImageCreate](#) is used to explicitly create a new image using encoded data does *PDEImageDataIsEncoded* return *true*.

Parameters

image

Image to examine.

encodedLenP

(Filled by the method) Length of the encoded data—if the data is encoded, that is, the method returns *true*.

Read/Write

Read

Return Value

true if [PDEImageGetData](#) returns encoded data, *false* otherwise. Returns *false* for *XObject* images.

Exceptions

[peErrWrongPDEObjectType](#)

Notifications

None

Header File

PERCalls.h

Related Methods

[PDEImageGetData](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDEImageGetAttrs

```
void PDEImageGetAttrs (PDEImage image, PDEImageAttrsP attrsP,
    ASUns32 attrsSize);
```

| | |
|------------------------|--|
| Description | Obtains the attributes for an image. |
| Parameters | <p><i>image</i></p> <p>Image whose attributes are obtained.</p> <p><i>attrsP</i></p> <p>Pointer to PDEImageAttrs structure with attributes of <i>image</i>.</p> <p><i>attrsSize</i></p> <p>Size of the <i>attrsP</i> buffer, in bytes.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrUnknownPDEColorSpace |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEImageGetData |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEImageGetColorSpace

[PDEColorSpace](#) PDEImageGetColorSpace ([PDEImage](#) image);

Description

Obtains the color space object for an image.

Use only for images. For image masks, use [PDEElementGetGState](#) to obtain color information.

Note: If you get the [PDEColorSpace](#) for an inline image, then get the CosObj for that color space with [PDEColorSpaceGetCosObj](#), this CosObj is limited. Cos objects that are the result of parsing inline dictionaries in the PDF page contents are a special class of Cos objects. You should never depend on these objects lasting the lifetime of the document. You should extract the information you need from the object immediately and refer to it no further in your code.

Parameters

image

Image whose color space is obtained.

Read/Write

Read

Return Value

Color space for *image*. Returns *NULL* if *image* is an image mask.

Exceptions

None

Notifications

None

Header File

PERCalls.h

Related Methods

None

Availability

Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEImageGetCosObj

```
void PDEImageGetCosObj (PDEImage image, CosObj* cosObjP);
```

| | |
|------------------------|--|
| Description | Obtains a Cos object for an image—if it is based on a Cos object. |
| Parameters | <p><i>image</i></p> <p>Image whose Cos object is obtained.</p> <p><i>cosObjP</i></p> <p>(Filled by the method) Cos object for the image. Returns the <i>NULL</i> Cos object if <i>image</i> is not Cos object based.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEImageCreateFromCosObj PDEImageIsCosObj |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEImageGetData

```
void PDEImageGetData (PDEImage image, ASUns32 flags,  
    ASUns8* buffer);
```

Description

Gets an image's data.

If the image is an *XObject* image, data is always returned as decoded data.

See the note about inline images under [PDEImageDatalsEncoded](#).

Parameters

image

Image whose data is obtained.

flags

Unused—must be zero.

buffer

Image data. If the data is decoded, *buffer* must be large enough to contain the number of bytes specified in the [PDEImageAttrs](#) structure obtained by [PDEImageGetAttrs](#). If the data is encoded, *buffer* must be large enough to contain the number of bytes in the *encodedLenP* parameter obtained by [PDEImageDatalsEncoded](#).

Read/Write

Read

Return Value

None

Exceptions

[peErrUnknownPDEColorSpace](#)
[genErrBadParm](#)
[peErrWrongPDEObjectType](#)

Notifications

None

Header File

PERCalls.h

Related Methods

[PDEImageDatalsEncoded](#)
[PDEImageSetData](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDEImageGetDataLen

```
ASInt32 PDEImageGetDataLen (PDEImage image);
```

| | |
|------------------------|---|
| Description | Obtains the length of data for an image. |
| Parameters | <p><i>image</i></p> <p>Image whose data length is obtained.</p> |
| Read/Write | Read |
| Return Value | Number of bytes of image data, specified by the width, height, bits per component, and color space of the image. |
| Exceptions | peErrUnknownPDEColorSpace genErrBadParm peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEImageGetData |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEImageGetDataStm

[ASStm](#) PDEImageGetDataStm ([PDEImage](#) image, ASUns32 flags);

Description Gets a data stream for an image. May only be called for *XObject* images.

The caller must dispose of the returned *ASStm* by calling [ASStmClose](#).

Parameters *image*

Image whose data stream is obtained.

flags

[PDEImageDataFlags](#) flags. If the *kPDEImageEncodedData* flag is set, data is returned in encoded form. Otherwise, data is decoded.

Read/Write Read

Return Value Stream for *image*.

Exceptions [peErrWrongPDEObjectType](#)
[genErrBadParm](#)

Notifications None

Header File *PERCalls.h*

Related Methods [PDEImageSetDataStm](#)

Availability Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEImageGetDecodeArray

```
ASUns32 PDEImageGetDecodeArray (PDEImage image,
    ASFixed* decode, ASUns32 decodeSize);
```

| | |
|------------------------|--|
| Description | Gets the decode array of from the attributes of the image. This array specifies the parameters used with the array of filters used to decode the image. |
| Parameters | <p><i>image</i></p> <p>The image whose decode array is obtained.</p> <p><i>decode</i></p> <p>(Filled by the method) Pointer to the decode array. If <i>NULL</i>, the number of decode elements required is returned.</p> <p><i>decodeSize</i></p> <p>Size of <i>decode</i> in bytes.</p> |
| Read/Write | Read |
| Return Value | Number of elements in the decode array. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEImageSetDecodeArray |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDEImageGetFilterArray

```
ASInt32 PDEImageGetFilterArray (PDEImage image,  
    PDEFilterArrayP filtersP);
```

| | |
|------------------------|--|
| Description | Obtains the filter array for an image. |
| Parameters | <i>image</i> Image whose filter array is obtained. <i>filtersP</i> (Filled by the method) Pointer to PDEFilterArray structure to fill with the current filter array for the image. <i>filtersP</i> must be large enough to contain all of the elements. May be <i>NULL</i> to obtain the number of filter elements. |
| Read/Write | Read |
| Return Value | Number of filter elements. |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEImageIsCosObj

```
ASBool PDEImageIsCosObj (PDEImage image);
```

| | |
|------------------------|--|
| Description | Determines if an image is represented by a Cos object. |
| Parameters | <i>image</i> Image to examine. |
| Read/Write | Read |
| Return Value | <i>true</i> if the image is represented by a Cos object, <i>false</i> otherwise. Returns <i>true</i> for <i>XObject</i> images and <i>false</i> for inline images. |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEImageGetCosObj |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEImageSetData

```
void PDEImageSetData (PDEImage image, ASUns32 flags,
    ASUns8* buffer, ASUns32 encodedLen);
```

| | |
|------------------------|--|
| Description | Sets data for an image. |
| Parameters | <p><i>image</i></p> <p>Image whose data is set.</p> <p><i>flags</i></p> <p>A set of PDEImageDataFlags flags. If <i>kPDEImageEncodedData</i> is set, the data must be encoded for the current filters, and <i>encodedLen</i> is the length of the encoded data.</p> <p>If the <i>kPDEImageEncodedData</i> flag is not set, data is not encoded and <i>encodedLen</i> is the size of the decoded data.</p> <p><i>buffer</i></p> <p>Image data.</p> <p><i>encodedLen</i></p> <p>Length of the encoded data.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrUnknownPDEColorSpace genErrBadParm peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEImageGetData |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDEImageSetDataStm

```
void PDEImageSetDataStm (PDEImage image, ASUns32 flags,
PDEFilterArrayP filtersP, ASStm stm);
```

| | |
|------------------------|---|
| Description | <p>Sets a data stream for an image. Can only be used for <i>XObject</i> images.</p> <p>The caller must dispose of the stream by calling ASStmClose.</p> |
| Parameters | <p><i>image</i></p> <p>Image whose data stream is set.</p> <p><i>flags</i></p> <p>PDEImageDataFlags flags. If the <i>kPDEImageEncodedData</i> flag is set, the stream must be encoded.</p> <p><i>filtersP</i></p> <p>Pointer to PDEFilterArray structure. If not <i>NULL</i>, is used to build Cos objects for the Filter, DecodeParms, and EncodeParms objects. If <i>filtersP</i> is <i>NULL</i>, the existing Filter and DecodeParms are used. EncodeParms is set to DecodeParms if it exists.</p> <p><i>stm</i></p> <p>Stream for the image data.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrUnknownPDEColorSpace peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEImageGetDataStm |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |
| Available in: | |

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEImageSetDecodeArray

```
void PDEImageSetDecodeArray (PDEImage image, ASFixed* decode,  
                             ASUns32 decodeSize);
```

Description Sets the decode array of the image. This array specifies the parameters used with the array of filters used to decode the image.

Normally, the decode array is accessed through the *decode* field in the [PDEImageAttrs](#) structure. However, this method defines a decode array to handle images with a colorspace that has 4 or less components.

Parameters *image*
Image whose decode array is set.

decode
Pointer to the decode array.

decodeSize
Size of *decode* array in bytes.

Read/Write Write

Return Value None

Exceptions None

Notifications None

Header File *PEWCalls.h*

Related Methods [PDEImageGetDecodeArray](#)

Availability Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEObject

PDEAcquire

```
void PDEAcquire ( PDEObject obj );
```

| | |
|------------------------|---|
| Description | Increments the reference count for an object. |
| Parameters | <p><i>obj</i></p> <p>The element whose count is incremented.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDERelease |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEAddTag

```
void PDEAddTag (PDEObject object, ASInt32 clientID,
               ASUns32 tag, void* value);
```

Description

Adds an identifier–value pair to an object.

The *clientID*–*tag* combination is a unique identifier for the value. Each client has its own identifier space. It is often convenient to use *ASAtoms* as tags.

Parameters

object

The element to tag. *object* may be a [PDEElement](#), [PDEContent](#), [PDEFont](#), [PDEColorSpace](#), and so on.

clientID

Identifies the caller/client. If there is only one client of the PDFEdit subsystem, *clientID* should be zero. If there are multiple clients, each should specify a nonzero, non-negative *clientID*. (A negative *clientID* is reserved for the implementation.)

tag

The tag to add to *object*. If *tag* is 0, this is the same as calling [PDERemoveTag](#). In other words, you can't tell the difference between a tag whose value is zero and a tag that is nonexistent.

value

Pointer to a value to associate with *object*. Only the pointer is stored. If the pointer points to data, it is the responsibility of the client to manage the data and its memory.

Read/Write

Write

Return Value

None

Exceptions

[peErrWrongPDEObjectType](#)

Notifications

None

Header File

PEWCalls.h

Related Methods

[PDEGetTag](#)
[PDERemoveTag](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEGetTag

```
void* PDEGetTag (PDEObject object, ASInt32 clientID,
                ASUns32 tag);
```

| | |
|------------------------|---|
| Description | Obtains an object's value for a given <i>clientID-tag</i> identifier that was added by PDEAddTag . |
| Parameters | <p><i>object</i></p> <p>The element whose value is obtained.</p> <p><i>clientID</i></p> <p>(Filled by the method) Client ID. Each client has its own identifier space.</p> <p><i>tag</i></p> <p>(Filled by the method) The <i>object's</i> tag. If <i>object</i> has no tag, this is 0.</p> |
| Read/Write | Write |
| Return Value | The value associated with the <i>clientID-tag</i> identifier. |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEAddTag PDERemoveTag |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEObjectGetType

```
ASInt32 PDEObjectGetType (PDEObject obj);
```

| | |
|------------------------|---|
| Description | Obtains the type of an element. |
| Parameters | <p><i>obj</i></p> <p>The element whose type is obtained.</p> |
| Read/Write | Read |
| Return Value | The object type, which is one of PDEType . |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDERelease

```
void PDERelease ( PDEObject obj );
```

| | |
|------------------------|---|
| Description | Decrements the reference count for the object. If the count becomes zero, the object is destroyed. Do not call <i>PDERelease</i> on <i>PDEContent</i> that you acquired with PDPageAcquirePDEContent ; call PDPageReleasePDEContent instead. |
| Parameters | <i>obj</i> The element released. |
| Read/Write | Read |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEAcquire |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDERemoveTag

```
void PDERemoveTag (PDEObject object, ASInt32 clientID,  
    ASUns32 tag);
```

Description Removes an object's value for a given *clientID*-*tag* identifier that was added by [PDEAddTag](#).
If [PDEAddTag](#) is called with a 0 tag, this is the same as calling *PDERemoveTag*.

Parameters

object
The element whose tag is removed.

clientID
Client ID. Each client has its own identifier space.

tag
Tag.

Read/Write Write

Return Value None

Exceptions [peErrWrongPDEObjectType](#)

Notifications None

Header File *PEWCalls.h*

Related Methods [PDEAddTag](#)
[PDEGetTag](#)

Availability Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEPath

PDEPathAddSegment

```
void PDEPathAddSegment (PDEPath path, ASUns32 segType,
    ASFixed x1, ASFixed y1, ASFixed x2, ASFixed y2, ASFixed x3,
    ASFixed y3);
```

Description

Adds a segment to a path. The number of *ASFixed* values used depends upon *segType*:

kPDEMoveTo: x1, y1

kPDELineTo: x1, y1

kPDECurveTo: x1, y1, x2, y2, x3, y3

kPDECurveToV: x1, y1, x2, y2

kPDECurveToY: x1, y1, x2, y2

kPDERect: x1, y1, x2 (width), y2 (height)

kPDEClosePath: None

Parameters

path

The path to which a segment is added.

data

A [PDEPathElementType](#) value indicating the type of path to add.

x1

x-coordinate of first point.

y1

y-coordinate of first point.

x2

x-coordinate of second point.

y2

y-coordinate of second point.

x3

x-coordinate of third point.

y3

y-coordinate of third point.

Read/Write

Write

Return Value

None

Exceptions [genErrBadParm](#)

Notifications None

Header File *PEWCalls.h*

Related Methods [PDEPathSetData](#)

Availability Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDEPathCreate

PDEPath PDEPathCreate (void);

| | |
|------------------------|--|
| Description | Creates an empty path element. |
| Parameters | None |
| Read/Write | Write |
| Return Value | An empty path element. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEPathGetData

```
ASUns32 PDEPathGetData (PDEPath path, ASInt32* data,
    ASUns32 dataSize);
```

| | |
|------------------------|--|
| Description | Obtains the size of the path data and, optionally, the path data. |
| Parameters | <p><i>path</i></p> <p>The path whose data is obtained.</p> <p><i>data</i></p> <p>(Filled by the method) Pointer to path data. If <i>data</i> is non-NULL, it contains a variable-sized array of path operators and operands. The format is a 32-bit operator followed by 0 to 3 ASFixedPoint values, depending on the operator. Opcodes are codes for moveto, lineto, curveto, rect, or closepath operators; operands are ASFixedPoint values.</p> <p>If <i>data</i> is NULL, the number of bytes required for data is returned by the method.</p> <p><i>dataSize</i></p> <p>Specifies the size of the buffer provided in <i>data</i>. If it is less than the length of the path data, the method copies <i>datasize</i> bytes.</p> |
| Read/Write | Read |
| Return Value | Length of data of <i>path</i> . |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEPathSetData |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDEPathGetPaintOp

```
ASUns32 PDEPathGetPaintOp (PDEPath path);
```

| | |
|------------------------|---|
| Description | Obtains the fill and stroke attributes of a path. |
| Parameters | <p><i>path</i></p> <p>The path whose fill and stroke attributes are obtained.</p> |
| Read/Write | Read |
| Return Value | A set of PDEPathOpFlags flags describing fill and stroke attributes. |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEPathSetPaintOp |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEPathSetData

```
void PDEPathSetData (PDEPath path, ASInt32* data,
    ASUns32 dataSize);
```

| | |
|------------------------|--|
| Description | Sets new path data for a path element. |
| Parameters | <p><i>path</i></p> <p>The path whose data is set.</p> <p><i>data</i></p> <p>Pointer to path data. It is a variable-sized array of path operators and operands. The format is a 32-bit operator followed by 0 to 3 ASFixedPoint values, depending on the operator. Operators are codes for moveto, lineto, curveto, rect, or closepath operators and must be one of PDEPathElementType. Operands are ASFixedPoint values.</p> <p><i>dataSize</i></p> <p>Size of the new path data, in bytes.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEPathGetData |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEPathSetPaintOp

```
void PDEPathSetPaintOp (PDEPath path, ASUns32 op);
```

| | |
|------------------------|---|
| Description | Sets the fill and stroke attributes of a path. |
| Parameters | <p><i>path</i></p> <p>The path whose fill and stroke attributes are set.</p> <p><i>op</i></p> <p>The operation to set; must be one of PDEPathOpFlags.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEPathGetPaintOp |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEPattern

PDEPatternCreate

[PDEPattern](#) PDEPatternCreate (const [CosObj](#)* cosObjP);

Description Creates a pattern object that can be used for a Pattern color space. See Section 7.16 in the [Portable Document Format Reference Manual](#) for more information about patterns.

Parameters *cosObjP*
A *CosStream* for the pattern.

Read/Write Write

Return Value A pattern.

Exceptions None

Notifications None

Header File *PEWCalls.h*

Related Methods [PDEPatternGetCosObj](#)

Availability Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEPatternGetCosObj

```
void PDEPatternGetCosObj (PDEPattern pattern, CosObj* cosObjP);
```

Description Obtains a Cos object corresponding to a pattern object.

Parameters *pattern*

Pattern whose Cos object is obtained.

cosObjP

(Filled by the method) Cos object for the pattern.

Read/Write Read

Return Value None

Exceptions None

Notifications None

Header File *PERCalls.h*

Related Methods [PDEPatternCreate](#)

Availability Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEPlace

PDEPlaceCreate

```
PDEPlace PDEPlaceCreate (ASAtom mcTag, CosObj* cosObjP,  
ASBool isInline);
```

| | |
|------------------------|---|
| Description | Creates a place object. |
| Parameters | <i>mcTag</i> Tag name for the place. <i>cosObjP</i> Optional Marked Content dictionary associated with the place. <i>isInline</i> If <i>true</i> , place is emitted into the page content stream inline. |
| Read/Write | Write |
| Return Value | The place object. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEPlaceGetDict

```
ASBool PDEPlaceGetDict (PDEPlace pdePlace, CosObj* placeDictP,
    ASBool* isInline);
```

| | |
|------------------------|---|
| Description | Obtains the Marked Content dictionary for a <i>PDEPlace</i> . |
| Parameters | <p><i>pdePlace</i></p> <p>The place whose Marked Content dictionary is obtained.</p> <p><i>placeDictP</i></p> <p>(Filled by the method) Pointer to the Marked Content dictionary; may be <i>NULL</i>.</p> <p><i>isInline</i></p> <p>(Filled by the method) If <i>true</i>, the Marked Content dictionary is inline; may be <i>NULL</i>.</p> |
| Read/Write | Read |
| Return Value | <i>true</i> if dictionary is obtained, <i>false</i> if no dictionary is present. |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEPlaceSetDict |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEPlaceGetMCTag

```
ASAtom PDEPlaceGetMCTag (PDEPlace pdePlace);
```

| | |
|------------------------|---|
| Description | Obtains the Marked Content tag for a <i>PDEPlace</i> . |
| Parameters | <i>pdePlace</i> The place whose Marked Content tag is obtained. |
| Read/Write | Read |
| Return Value | Tag for <i>pdePlace</i> . |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEPlaceSetMCTag |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEPlaceSetDict

```
void PDEPlaceSetDict (PDEPlace pdePlace, CosObj* placeDictP,  
    ASBool isInline);
```

| | |
|------------------------|---|
| Description | Sets the Marked Content dictionary for a <i>PDEPlace</i> . |
| Parameters | <i>pdePlace</i> The place whose Marked Content dictionary is set. <i>placeDictP</i> Marked Content dictionary for <i>pdePlace</i> . <i>isInline</i> If <i>true</i> , the dictionary is emitted inline. |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEPlaceGetDict |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEPlaceSetMCTag

```
void PDEPlaceSetMCTag (PDEPlace pdePlace, ASAtom mcTag);
```

| | |
|------------------------|--|
| Description | Sets the Marked Content tag for a <i>PDEPlace</i> . |
| Parameters | <p><i>pdePlace</i> The place whose Marked Content tag is set.</p> <p><i>mcTag</i> The tag for <i>pdePlace</i>.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEPlaceGetMCTag |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEShading

PDEShadingCreateFromCosObj

```
PDEShading PDEShadingCreateFromCosObj (const CosObj* shadingP,  
    ASFixedMatrixP matrixP);
```

| | |
|------------------------|--|
| Description | Creates a smooth shading object. |
| Parameters | <p><i>shadingP</i></p> <p>The shading dictionary.</p> <p><i>matrixP</i></p> <p>The location and transformation matrix of the shading object.</p> |
| Return Value | A smooth shading object. |
| Exceptions | peErrUnknownPDEColorSpace cosErrInvalidObj cosErrExpectedName genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEShadingGetCosObj

```
void PDEShadingGetCosObj (PDEShading shading, CosObj* cosObjP);
```

| | |
|------------------------|--|
| Description | Gets the <i>CosObj</i> for a <i>PDEShading</i> . |
| Parameters | <p><i>shading</i> A smooth shading object.</p> <p><i>cosObjP</i> The Cos dictionary corresponding to <i>shading</i>.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEText

PDETextAdd

```
void PDETextAdd (PDEText pdeText, ASUns32 flags, ASInt32 index,
    ASUns8* text, ASInt32 textLen, PDEFont font,
    PDEGraphicStateP gstateP, ASUns32 gstateLen,
    PDETextStateP tstateP, ASUns32 tstateLen,
    ASFixedMatrixP textMatrixP, ASFixedMatrixP strokeMatrixP);
```

Description

Adds a character or a text run to a *PDEText* object.

Parameters

pdeText

Text object to which a character or text run is added.

flags

A [PDETextFlags](#) that specifies what kind of text to add. Must be either:

kPDETextChar — for a text character

kPDETextRun — for a text run

index

Index after which to add character or text run.

text

Pointer to the characters to add.

Note: Passing NULL for text can invalidate text object but will not raise an error. Callers should avoid passing NULL for this parameter.

textLen

Length of the text, in bytes.

Note: Passing NULL for textLen can invalidate text object but will not raise an error. Callers should avoid passing NULL for this parameter.

font

Font for the element.

gstateP

Pointer to a [PDEGraphicState](#) structure with the graphics state for the element.

gstateLen

Length of graphics state for the element.

tstateP

Pointer to a [PDETextState](#) structure with text state for the element.

tstateLen

Length of text state for the element.

textMatrixP

Pointer to [ASFixedMatrix](#) that holds the matrix for the element.

strokeMatrixP

Pointer to [ASFixedMatrix](#) that holds the matrix for the line width when stroking text.

Read/Write Write

Return Value None

Exceptions [pdErrBadResMetrics](#)
[peErrWrongPDEObjectType](#)
[genErrBadParm](#)

Notifications None

Header File *PEWCalls.h*

Related Methods [PDETextIsAtPoint](#)
[PDETextReplaceChars](#)
[PDETextSplitRunAt](#)

Availability Plug-ins: Available if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextCreate

PDEText PDETextCreate (void);

| | |
|------------------------|--|
| Description | Creates an empty text object. |
| Parameters | None |
| Read/Write | Write |
| Return Value | An empty text object. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextGetAdvanceWidth

```
void PDETextGetAdvanceWidth (PDEText pdeText, ASUns32 flags,
    ASInt32 index, ASFixedPointP advanceP);
```

Description

Obtains the advance width of a character or a text element. Advance width is returned in either character space or page space. The advance width is the amount by which the current point advances when the character is drawn.

Advance width may be horizontal or vertical, depending on the writing style. Thus *advanceP* has both a horizontal and vertical component.

Parameters

pdeText

Text object containing a character or text run whose advance width is found.

flags

A [PDETextFlags](#) value that specifies whether *index* refers to the character offset from the beginning of the text object or the index of the text run in the text object. Must be either:

kPDETextChar — for a text character

kPDETextRun — for a text run

In addition, set the *kPDETextPageSpace* flag to obtain the advance width in page space. If it is not set, the advance width is in character space.

index

Index of the character or text run in *pdeText*.

advanceP

(Filled by the method) Pointer to a [ASFixedPoint](#) value indicating the advance width.

Read/Write

Read

Return Value

None

Exceptions

[peErrWrongPDEObjectType](#)
[genErrBadParm](#)
[pdErrBadResMetrics](#)

Notifications

None

Header File

PERCalls.h

Related Methods None

Availability Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextGetBBox

```
void PDETextGetBBox (PDEText pdeText, ASUns32 flags,
    ASInt32 index, ASFixedRectP bboxP);
```

Description Obtains the bounding box of a character or a text run.

Parameters

pdeText

Text object containing a character or text run whose bounding box is found.

flags

A [PDETextFlags](#) that specifies whether *index* refers to the character offset from the beginning of the text object or the index of the text run in the text object. Must be either:

kPDETextChar — for a text character

kPDETextRun — for a text run

index

Index of the character or text run in *pdeText*.

bboxP

(Filled by the method) Pointer to [ASFixedRect](#) to set to the bounding box of specified character or text run.

Read/Write Read

Return Value None

Exceptions [peErrWrongPDEObjectType](#)
[genErrBadParm](#)
[pdErrBadResMetrics](#)

Notifications None

Header File *PERCalls.h*

Related Methods None

Availability Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDETextGetFont

```
PDEFont PDETextGetFont (PDEText pdeText, ASUns32 flags,  
ASInt32 index);
```

| | |
|------------------------|--|
| Description | Obtains the font for a text character or element. |
| Parameters | <p><i>pdeText</i></p> <p>Text object containing a character or text run whose font is found.</p> <p><i>flags</i></p> <p>A PDETextFlags that specifies whether <i>index</i> refers to the character offset from the beginning of the text object or the index of the text run in the text object. Must be either:</p> <p><i>kPDETextChar</i> — for a text character</p> <p><i>kPDETextRun</i> — for a text run</p> <p><i>index</i></p> <p>Index of the character or text run in <i>pdeText</i>.</p> |
| Read/Write | Read |
| Return Value | Font of the specified character or text run. |
| Exceptions | peErrWrongPDEObjectType genErrBadParm pdErrBadResMetrics |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDETextRunSetFont |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | — | — |

PDETextGetGState

```
void PDETextGetGState (PDEText pdeText, ASUns32 flags,
    ASInt32 index, PDEGraphicStateP stateP, ASUns32 stateSize);
```

| | |
|------------------------|---|
| Description | Obtains the graphics state of a character or a text run. |
| Parameters | <p><i>pdeText</i></p> <p>Text object containing a character or text run whose graphics state is found.</p> <p><i>flags</i></p> <p>A PDETextFlags that specifies whether <i>index</i> refers to the character offset from the beginning of the text object or the index of the text run in the text object. Must be either:</p> <p><i>kPDETextChar</i> — for a text character</p> <p><i>kPDETextRun</i> — for a text run</p> <p><i>index</i></p> <p>Index of the character or text run in <i>pdeText</i>.</p> <p><i>stateP</i></p> <p>(Filled by the method) Pointer to a PDEGraphicState structure with graphics state of specified character or text run.</p> <p><i>stateSize</i></p> <p>Size of the <i>stateP</i> buffer, in bytes.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDETextRunSetGState |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |
| Available in: | |

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextGetNumBytes

```
ASInt32 PDETextGetNumBytes (PDEText pdeText, ASUns32 flags,  
    ASInt32 index);
```

| | |
|------------------------|---|
| Description | Gets the number of bytes occupied by the character code or text run. |
| Parameters | <p><i>pdeText</i></p> <p>A <i>PDEText</i> object returned from one of the PDETextCreate methods whose text is examined.</p> <p><i>flags</i></p> <p>A PDETextFlags that specifies whether <i>index</i> refers to the character offset from the beginning of the text object or the index of the text run in the text object. Must be either:</p> <p><i>kPDETextChar</i> — for a text character</p> <p><i>kPDETextRun</i> — for a text run</p> <p><i>index</i></p> <p>Index of the character or text run in <i>pdeText</i>.</p> |
| Return Value | Number of bytes occupied by the text run or character. |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEFontGetNumCodeBytes |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextGetNumChars

```
ASInt32 PDETextGetNumChars ( PDEText pdeText );
```

| | |
|------------------------|---|
| Description | Obtains the number of characters in a text object. |
| Parameters | <i>pdeText</i> Text object whose number of characters is found. |
| Read/Write | Read |
| Return Value | Total number of characters in <i>pdeText</i> . |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDETextGetNumRuns PDETextGetRunForChar |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextGetNumRuns

```
ASInt32 PDETextGetNumRuns ( PDEText pdeText ) ;
```

| | |
|------------------------|---|
| Description | Obtains the number of text runs (show strings) in a text object. |
| Parameters | <i>pdeText</i> Text object whose number of text runs is found. |
| Read/Write | Read |
| Return Value | Number of text runs in <i>pdeText</i> . |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDETextGetNumBytes PDETextGetRunForChar |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextGetQuad

```
void PDETextGetQuad (PDEText pdeText, ASUns32 flags,
    ASInt32 index, ASFixedQuadP quadP);
```

| | |
|----------------------|--|
| Description | <p>Obtains the quad bounding the specified text run or character.</p> <p>The advance portion of the quad is based on the left sidebearing and advance width.</p> |
| Parameters | <p><i>pdeText</i></p> <p>Text object containing a character or text run whose quad is found.</p> <p><i>flags</i></p> <p>A PDETextFlags that specifies whether <i>index</i> refers to the character offset from the beginning of the text object or the index of the text run in the text object. Must be either:</p> <p><i>kPDETextChar</i> — for a text character</p> <p><i>kPDETextRun</i> — for a text run</p> <p>In addition, if the <i>kPDETextBounding</i> flag is set, <i>PDETextGetQuad</i> uses the font descriptor's FontBBox, which is the smallest rectangle that encloses all characters in the font. The advance portion is based on the x-coordinates of the left and right sides of FontBBox and the advance width.</p> <p><i>index</i></p> <p>Index of the character or text run in <i>pdeText</i>.</p> <p><i>quadP</i></p> <p>(Filled by the method) Pointer to ASFixedQuad that bounds the specified character or text run.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm pdErrBadResMetrics |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |

Related Methods None

Availability Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextGetRunForChar

```
ASInt32 PDETextGetRunForChar ( PDEText pdeText ,
    ASInt32 charIndex );
```

| | |
|------------------------|--|
| Description | Obtains the index of the text run that contains the n th character in a text object. |
| Parameters | <p><i>pdeText</i> Text object to examine.</p> <p><i>charIndex</i> Number of the character to find in <i>pdeText</i>.</p> |
| Read/Write | Read |
| Return Value | Index of the text run with the specified character index into <i>pdeText</i> . |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextGetStrokeMatrix

```
void PDETextGetStrokeMatrix (PDEText pdeText, ASUns32 flags,
    ASInt32 index, ASFixedMatrixP matrixP);
```

| | |
|------------------------|--|
| Description | Obtains the stroke matrix of a character or a text run. |
| Parameters | <p><i>pdeText</i></p> <p>Text object containing a character or text run whose stroke matrix is found.</p> <p><i>flags</i></p> <p>A PDETextFlags that specifies whether <i>index</i> refers to the character offset from the beginning of the text object or the index of the text run in the text object. Must be either:</p> <p><i>kPDETextChar</i> — for a text character</p> <p><i>kPDETextRun</i> — for a text run</p> <p><i>index</i></p> <p>Index of the character or text run in <i>pdeText</i>.</p> <p><i>matrixP</i></p> <p>(Filled by the method) Pointer to ASFixedMatrix that holds the stroke matrix of specified character or text run. This matrix is the transformation for line widths when stroking. The <i>h</i> and <i>v</i> values of the matrix are ignored.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDETextRunSetStrokeMatrix |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |
| Available in: | |

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextGetText

```
ASInt32 PDETextGetText (PDEText pdeText, ASUns32 flags,
    ASInt32 index, ASUns8* textBuffer);
```

Description

Gets the text for a text run or character.

Parameters

pdeText

Text object containing a character or text run whose text is found.

flags

A [PDETextFlags](#) that specifies whether *index* refers to the character offset from the beginning of the text object or the index of the text run in the text object. Must be either:

kPDETextChar — for a text character

kPDETextRun — for a text run

index

Index of the character or text run in *pdeText*.

textBuffer

(Filled by the method) Text of specified character or text run. *textBuffer* must be large enough to hold the returned text.

If *textBuffer* is *NULL*, returns the number of bytes required to hold the data.

Read/Write

Read

Return Value

Number of bytes in text run or character.

Exceptions

[peErrWrongPDEObjectType](#)
[genErrBadParm](#)

Notifications

None

Header File

PERCalls.h

Related Methods

[PDETextGetTextMatrix](#)
[PDETextGetTextState](#)

Availability

Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextGetTextMatrix

```
void PDETextGetTextMatrix (PDEText pdeText, ASUns32 flags,
    ASInt32 index, ASFixedMatrixP matrixP);
```

| | |
|------------------------|---|
| Description | Obtains the matrix of a character or a text run. |
| Parameters | <p><i>pdeText</i></p> <p>Text object containing a character or text run whose matrix is found.</p> <p><i>flags</i></p> <p>A PDETextFlags that specifies whether <i>index</i> refers to the character offset from the beginning of the text object or the index of the text run in the text object. Must be either:</p> <p><i>kPDETextChar</i> — for a text character</p> <p><i>kPDETextRun</i> — for a text run</p> <p><i>index</i></p> <p>Index of the character or text run in <i>pdeText</i>.</p> <p><i>matrixP</i></p> <p>(Filled by the method) Pointer to ASFixedMatrix that holds the matrix of specified character or text run. This is the transformation matrix from page space to the current text space. The <i>h</i> and <i>v</i> values of the matrix indicate the origin of the first character.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm pdErrBadResMetrics |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDETextRunSetTextMatrix |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |
| Available in: | |

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextGetTextState

```
void PDETextGetTextState (PDEText pdeText, ASUns32 flags,
    ASInt32 index, PDETextStateP stateP, ASUns32 stateSize);
```

| | |
|------------------------|--|
| Description | Obtains the text state of a character or a text run. |
| Parameters | <p><i>pdeText</i></p> <p>Text object containing a character or text run whose text state is found.</p> <p><i>flags</i></p> <p>A PDETextFlags that specifies whether <i>index</i> refers to the character offset from the beginning of the text object or the index of the text run in the text object. Must be either:</p> <p><i>kPDETextChar</i> — for a text character</p> <p><i>kPDETextRun</i> — for a text run</p> <p><i>index</i></p> <p>Index of the character or text run in <i>pdeText</i>.</p> <p><i>stateP</i></p> <p>(Filled by the method) Pointer to a PDETextState structure to fill with the text state of the specified character or text run.</p> <p><i>stateSize</i></p> <p>Size of the <i>stateP</i> buffer, in bytes.</p> |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDETextRunSetTextState |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |
| Available in: | |

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextIsAtPoint

```
ASBool PDETextIsAtPoint (PDEText pdeText, ASUns32 flags,  
    ASInt32 index, ASFixedPoint point);
```

Description Tests whether a point is on specified text. Checks if the point is in a bounding box for the *PDEText*.

Parameters *pdeText*

The text to test.

flags

A [PDETextFlags](#) that specifies whether *index* refers to the character offset from the beginning of the text object or the index of the text run in the text object. Must be either:

kPDETextChar — for a text character

kPDETextRun — for a text run

index

Index of the character or text run in *pdeText*.

point

The point, specified in user space coordinates.

Return Value *true* if the point is on the text, *false* otherwise.

Exceptions None

Notifications None

Header File *PERCalls.h*

Related Methods [PDEElementIsAtPoint](#)
[PDEElementIsAtRect](#)

Availability Plug-ins: Available if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | — | — |

PDETextIsAtRect

```
ASBool PDETextIsAtRect (PDEText pdeText, ASUns32 flags,  
    ASInt32 index, ASFixedRect rect);
```

| | |
|------------------------|---|
| Description | Tests whether any part of a rectangle is on the specified text. |
| Parameters | <p><i>pdeText</i></p> <p>The text to test.</p> <p><i>flags</i></p> <p>A PDETextFlags flag that specifies whether <i>index</i> refers to the character offset from the beginning of the text object or the index of the text run in the text object. Must be either:</p> <p><i>kPDETextChar</i> — for a text character</p> <p><i>kPDETextRun</i> — for a text run</p> <p><i>index</i></p> <p>Index of the character or text run in <i>pdeText</i>.</p> <p><i>rect</i></p> <p>The rectangle, specified in user space coordinates.</p> |
| Return Value | <i>true</i> if the text is on the rectangle, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEElementIsAtPoint PDEElementIsAtRect PDETextIsAtPoint |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDETextRemove

```
void PDETextRemove (PDEText pdeText, ASUns32 flags,
    ASInt32 index, ASInt32 count);
```

| | |
|------------------------|--|
| Description | Removes characters or text runs from a text object. |
| Parameters | <p><i>pdeText</i></p> <p>Text object from which text is removed.</p> <p><i>flags</i></p> <p>A PDETextFlags that specifies whether <i>index</i> refers to the character offset from the beginning of the text object or the index of the text run in the text object. Must be either:</p> <p><i>kPDETextChar</i> — for a text character</p> <p><i>kPDETextRun</i> — for a text run</p> <p><i>index</i></p> <p>Index of the character or text run in <i>pdeText</i>.</p> <p><i>count</i></p> <p>Number of characters or text runs to remove.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm pdErrBadResMetrics |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDETextAdd PDETextReplaceChars PDETextSplitRunAt |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDETextReplaceChars

```
void PDETextReplaceChars (PDEText pdeText, ASUns32 flags,
    ASInt32 index, ASUns8* textBuffer, ASInt32 numBytes);
```

| | |
|------------------------|---|
| Description | Replaces characters in a text object. This method does not change the number of characters in the text object—extra characters are ignored. |
| Parameters | <p><i>pdeText</i> Text object in which characters are replaced.</p> <p><i>flags</i> A PDETextFlags that specifies whether <i>index</i> refers to the character offset from the beginning of the text object or the index of the text run in the text object. Must be either: <i>kPDETextChar</i> — for a text character <i>kPDETextRun</i> — for a text run</p> <p><i>index</i> Index of the character or text run in <i>pdeText</i>.</p> <p><i>textBuffer</i> Replacement text.</p> <p><i>numBytes</i> Number of bytes to replace.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDETextAdd PDETextIsAtPoint PDETextSplitRunAt |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |
| Available in: | |

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextRunGetCharOffset

```
ASInt32 PDETextRunGetCharOffset (PDEText pdeText,
    ASInt32 runIndex);
```

| | |
|------------------------|---|
| Description | Obtains the character offset of the first character of the specified text run. |
| Parameters | <p><i>pdeText</i></p> <p>Text object containing a character or text run whose graphics state is found.</p> <p><i>runIndex</i></p> <p>Index of the text run whose first character's index is returned.</p> |
| Read/Write | Read |
| Return Value | Character offset of the first character of the specified text run in <i>pdeText</i> . |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDETextGetNumBytes PDETextGetNumRuns PDETextGetRunForChar |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextRunGetNumChars

```
ASInt32 PDETextRunGetNumChars (PDEText pdeText ,
    ASInt32 runIndex);
```

| | |
|------------------------|---|
| Description | Obtains the number of characters in a text run. |
| Parameters | <p><i>pdeText</i></p> <p>Text object containing a text run whose number of characters is found.</p> <p><i>runIndex</i></p> <p>Index of the text run whose number of characters is returned.</p> |
| Read/Write | Read |
| Return Value | Number of characters in the specified text run. |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDETextGetNumRuns PDETextGetRunForChar PDETextRunGetCharOffset |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextRunSetFont

```
void PDETextRunSetFont (PDEText pdeText, ASInt32 runIndex,  
    PDEFont font);
```

| | |
|------------------------|--|
| Description | Sets the font of a text run. |
| Parameters | <i>pdeText</i> Text object containing a text run whose font is set. <i>runIndex</i> Index of the text run. <i>font</i> Font set for the text run. |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDETextGetFont |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextRunSetGState

```
void PDETextRunSetGState (PDEText pdeText, ASInt32 runIndex,
PDEGraphicStateP stateP, ASUns32 stateSize);
```

| | |
|------------------------|--|
| Description | Sets the graphics state of a text run. |
| Parameters | <p><i>pdeText</i></p> <p>Text object containing a text run whose graphics state is set.</p> <p><i>runIndex</i></p> <p>Index of the text run.</p> <p><i>stateP</i></p> <p>Pointer to a PDEGraphicState structure with graphics state to set.</p> <p><i>stateSize</i></p> <p>Size of the <i>stateP</i> buffer, in bytes.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDETextGetGState |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextRunSetStrokeMatrix

```
void PDETextRunSetStrokeMatrix (PDEText pdeText,
    ASInt32 runIndex, ASFixedMatrixP matrixP);
```

| | |
|------------------------|---|
| Description | Sets the stroke matrix of a text run. |
| Parameters | <p><i>pdeText</i></p> <p>Text object containing a text run whose stroke matrix is set.</p> <p><i>runIndex</i></p> <p>Index of the text run.</p> <p><i>matrixP</i></p> <p>Pointer to ASFixedMatrix that holds the stroke matrix.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDETextGetStrokeMatrix |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextRunSetTextMatrix

```
void PDETextRunSetTextMatrix (PDEText pdeText,
    ASInt32 runIndex, ASFixedMatrixP matrixP);
```

| | |
|------------------------|---|
| Description | Sets the text matrix of a text run. |
| Parameters | <p><i>pdeText</i></p> <p>Text object containing a text run whose text matrix is set.</p> <p><i>runIndex</i></p> <p>Index of the text run.</p> <p><i>matrixP</i></p> <p>Pointer to ASFixedMatrix that holds the text matrix.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDETextGetTextMatrix |
| Availability | Plug-ins: Available if <i>PL_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextRunSetTextState

```
void PDETextRunSetTextState (PDEText pdeText, ASInt32 runIndex,
    PDETextStateP stateP, ASUns32 stateSize);
```

| | |
|------------------------|--|
| Description | Sets the text state of a text run. |
| Parameters | <p><i>pdeText</i></p> <p>Text object containing a text run whose text state is set.</p> <p><i>runIndex</i></p> <p>Index of the text run.</p> <p><i>stateP</i></p> <p>Pointer to a PDETextState structure with text state.</p> <p><i>stateSize</i></p> <p>Size of the <i>stateP</i> buffer, in bytes.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDETextGetTextState |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDETextSplitRunAt

```
void PDETextSplitRunAt (PDEText pdeText, ASInt32 splitLoc);
```

| | |
|------------------------|---|
| Description | Splits a text run into two text runs. |
| Parameters | <i>pdeText</i> Text object containing a text run to split. <i>splitLoc</i> Split location. <i>splitLoc</i> is relative to the text object. The first text run is from character index 0 up to <i>splitLoc</i> . The second text run is from <i>splitLoc</i> + 1 to the end of the run. |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType genErrBadParm pdErrBadResMetrics |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDETextIsAtPoint PDETextReplaceChars |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEUnknown

PDEUnknownGetOpName

```
ASAtom PDEUnknownGetOpName ( PDEUnknown pdeUnknown );
```

| | |
|------------------------|--|
| Description | Gets the operator name of an unknown operator. |
| Parameters | <p><i>pdeUnknown</i></p> <p>Unknown element whose operator name is obtained.</p> |
| Read/Write | Read |
| Return Value | An <i>ASAtom</i> for the name of the operator for <i>pdeUnknown</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEXObject

PDEXObjectCreate

```
PDEXObject PDEXObjectCreate (const CosObj* cosObjP);
```

| | |
|------------------------|--|
| Description | Creates a new <i>PDEXObject</i> from a Cos object. |
| Parameters | <i>cosObjP</i> Cos object for the <i>PDEXObject</i> . |
| Read/Write | Write |
| Return Value | <i>PDEXObject</i> corresponding to the <i>cosObjP</i> . |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PEWCalls.h</i> |
| Related Methods | PDEXObjectGetCosObj |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDEXObjectGetCosObj

```
void PDEXObjectGetCosObj (PDEXObject xObject, CosObj* cosObjP);
```

| | |
|------------------------|---|
| Description | Gets a Cos object corresponding to a <i>PDEXObject</i> . |
| Parameters | <i>xObject</i> The <i>PDEXObject</i> whose Cos object is obtained. <i>cosObjP</i> (Filled by the method) Cos object for <i>xObject</i> . |
| Read/Write | Read |
| Return Value | None |
| Exceptions | peErrWrongPDEObjectType |
| Notifications | None |
| Header File | <i>PERCalls.h</i> |
| Related Methods | PDEXObjectCreate |
| Availability | Plug-ins: Available if <i>PI_PDFEDIT_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSysFont

PDEmbedSysFontForPDEFont

```
void PDEmbedSysFontForPDEFont (PDEFont font, ASUns32 flags,  
    CosDoc cosDoc);
```

Description

If there is a font on the system that matches this *PDEFont*, embed the full font, regardless of whether it was subsetting or not embedded at all in the first place. This will not work for CID fonts, because they must be subsetting.

The matching is based on the [PDSysFontMatchFlags](#).

Parameters

font

A *PDEFont* object returned from one of the [PDEFontCreate](#) methods.

flags

Flags from [PDSysFontMatchFlags](#) that determine matches.

cosDoc

Currently unused.

Return Value

None

Exceptions

Raises [peErrFontToEmbedNotOnSys](#) if there is no system font that matches this *PDEFont*.

Raises [genErrBadParm](#) if the *PDEFont* is a CID font.

[peErrCantCreateFontSubset](#)

[peErrCantGetAttrs](#)

[peErrCantGetWidths](#)

Notifications

None

Header File

PSFCalls.h

Related Methods

[PDEFontCreateFromSysFont](#)

[PDFindSysFontForPDEFont](#)

Availability

Plug-ins: Available if *PI_PDSYSFONT_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|---------|
| Adobe PDF Library | — | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDEnumSysFonts

```
void PDEnumSysFonts (PDSysFontEnumProc enumProc,
    void* clientData);
```

Description

Enumerates all of the system fonts with a user-supplied procedure.

The [PDSysFont](#) may be acquired during the enumeration.

Developers should *not* assume that the *enumProc* will get called. If no system fonts are found (for example, if the *PSRESOURCEPATH* environment variable is not set on UNIX platforms), *enumProc* is never called, and *PDEnumSysFonts* does not raise an exception.

Parameters

enumProc

User-supplied callback to call once for each system font.

Enumeration continues until all fonts have been enumerated, or until *enumProc* returns *false*.

clientData

Pointer to user-supplied data to pass to *enumProc* each time it is called.

Read/Write

Write

Return Value

None

Exceptions

None

Notifications

None

Header File

PSFCalls.h

Related Methods

[PDFFindSysFont](#)
[PDFFindSysFontForPDEFont](#)

Availability

Plug-ins: Available if *PI_PDSYSFONT_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDFindSysFont

```
PDSysFont PDFindSysFont (PDEFontAttrsP attrs, ASUns32 attrsSize,
    ASUns32 flags);
```

| | |
|------------------------|--|
| Description | <p>Finds a system font that matches the requested attributes.</p> <p>The method gets the <i>PDSysFont</i> rather than acquires it, so do <i>not</i> call PDERelease on the returned <i>PDSysFont</i> when done with it.</p> |
| Parameters | <p><i>attrs</i></p> <p>Pointer to a <u>PDEFontAttrs</u> structure with the attributes of the font you are searching for.</p> <p><i>attrsSize</i></p> <p>Size of the <i>attrs</i> buffer, in bytes.</p> <p><i>flags</i></p> <p>Flags from <u>PDSysFontMatchFlags</u>.</p> |
| Read/Write | Write |
| Return Value | The desired system font. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PSFCalls.h</i> |
| Related Methods | PDEnumSysFonts PDFindSysFontForPDEFont |
| Availability | Plug-ins: Available if <i>PI_PDSYSFONT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDFindSysFontEx

```
PDSysFont PDFindSysFontEx (PDEFontAttrsP attrs,
    ASUns32 attrsSize, ASUns32 flags, ASFixed* mmDesignVector,
    ASInt32* designVecLength);
```

| | |
|----------------------|---|
| Description | <p>Finds a system font that matches the requested attributes.</p> <p>If the requested font is a multiple master font instance, the base font is returned, and the specified design vector is decoded and returned in <i>mmDesignVector</i>.</p> <p>The method gets the <i>PDSysFont</i> rather than acquires it, so do <i>not</i> call <u>PDERelease</u> on the returned <i>PDSysFont</i> when done with it.</p> |
| Parameters | <p><i>attrs</i></p> <p>Pointer to a <u>PDEFontAttrs</u> structure with the attributes of the font you are searching for.</p> <p><i>attrsSize</i></p> <p>Size of the <i>attrs</i> buffer, in bytes.</p> <p><i>flags</i></p> <p>Flags from <u>PDSysFontMatchFlags</u>.</p> <p><i>mmDesignVector</i></p> <p>(Filled by the method) If the requested font is a multiple master font instance, the specified design vector is decoded and returned in <i>mmDesignVector</i>.</p> <p><i>designVecLength</i></p> <p>(Filled by the method) Pass the length of <i>mmDesignVector</i>. This parameter also returns the number of elements filled in <i>mmDesignVector</i> (maximum = 4).</p> |
| Read/Write | Write |
| Return Value | The desired system font. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PSFCalls.h</i> |

Related Methods [PDEnumSysFonts](#)
[PDFindSysFontForPDEFont](#)

Availability Plug-ins: Available if *PI_PDSYSFONT_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSysFontAcquirePlatformData

[PDSysFontPlatDataP](#) PDSysFontAcquirePlatformData
([PDSysFont](#) sysFont);

| | |
|------------------------|---|
| Description | Acquires platform-specific data for use by user interface code. Must be released when finished by PDSysFontReleasePlatformData . |
| Parameters | <i>sysFont</i> A <i>PDSysFont</i> object referencing a system font returned by either PDFindSysFont or PDFindSysFontForPDEFont . |
| Return Value | Pointer to a platform-dependent structure <i>PDSysFontPlatData</i> containing information relating to a system font. Returns <i>NULL</i> if out of memory. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PSFCalls.h</i> |
| Related Methods | PDSysFontReleasePlatformData |
| Availability | Plug-ins: Available if <i>PI_PDSYSFONT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSysFontGetAttrs

```
void PDSysFontGetAttrs (PDSysFont sysFont, PDEFontAttrsP attrsP,
    ASUns32 attrsSize);
```

Description

Obtains the attributes of a system font.

The attributes will be returned in the buffer pointed to by *attrsP*.

No more than *attrsSize* bytes will be written to the buffer.

This call can be expensive to execute, as it may involve parsing the font in order to determine attributes.

Parameters

sysFont

A *PDSysFont* object referencing a system font whose attributes are obtained.

attrsP

(Filled by the method) Pointer to a [PDEFontAttrs](#) structure with the attributes of a system font.

attrsSize

Size of the *attrsP* buffer, in bytes.

Read/Write

Write

Return Value

None

Exceptions

[peErrCantGetAttrs](#)

Notifications

None

Header File

PSFCalls.h

Related Methods

[PDSysFontGetEncoding](#)
[PDSysFontGetInfo](#)
[PDSysFontGetName](#)
[PDSysFontGetType0Widths](#)

Availability

Plug-ins: Available if *PI_PDSYSFONT_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDSysFontGetCIDSystemInfo

```
void PDSysFontGetCIDSystemInfo (PDSysFont sysFont,  
    ASAtom* registry, ASAtom* ordering, ASInt32* supplement);
```

| | |
|------------------------|--|
| Description | Derives the registry, ordering, and supplement information of a multibyte system font from the IfCharSet . This information can be used to create a <i>PDEFont</i> from a system font. For more information on CID fonts, see <i>PDFontGetCIDSystemInfo</i> . |
| Parameters | <p><i>sysFont</i></p> <p>A <i>PDSysFont</i> object referencing a multibyte system font.</p> <p><i>registry</i></p> <p>(Filled by the method) The <i>ASAtom</i> representing the CIDFont's Registry information, as in "Adobe-Japan".</p> <p><i>ordering</i></p> <p>(Filled by the method) The <i>ASAtom</i> representing the CIDFont's Ordering information, for example, "1".</p> <p><i>supplement</i></p> <p>(Filled by the method) The SystemSupplement field from the CIDFont.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PSFCalls.h</i> |
| Related Methods | PDFontGetCIDSystemInfo PDFontGetCIDSystemSupplement |
| Availability | Plug-ins: Available if <i>PI_PDSYSFONT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDSysFontGetEncoding

```
ASUns8** PDSysFontGetEncoding (PDSysFont sysFont,
    ASAtom* encodingNameP);
```

| | |
|------------------------|---|
| Description | Obtains the encoding of a single byte encoded system font. The returned encoding must be freed via a call to ASfree . |
| Parameters | <p><i>sysFont</i></p> <p>A <i>PDSysFont</i> object referencing a system font whose encoding is obtained.</p> <p><i>encodingNameP</i></p> <p>(Filled by the method) An encoding name if the return value of <i>PDSysFontGetEncoding</i> is zero.</p> <p>If <i>encodingNameP</i> is the <i>NULL ASAtom</i>, the font uses its default encoding.</p> |
| Read/Write | Write |
| Return Value | <p>An encoding array of 256 C strings. Each entry in the array either contains a glyph name or <i>NULL</i>. If it is <i>NULL</i>, the corresponding entry uses the font's built in encoding value.</p> <p>If the return value is zero, <i>encodingNameP</i> contains the name of the encoding.</p> <ul style="list-style-type: none"> For a Type 1 font, the default encoding is that specified by the Encoding value in the font dictionary. For a TrueType font, the default encoding is that specified in the single byte CMAP table. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PSFCalls.h</i> |
| Related Methods | PDSysFontAcquirePlatformData PDSysFontGetInfo PDSysFontGetName PDSysFontGetType0Widths |
| Availability | Plug-ins: Available if <i>PI_PDSYSFONT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSysFontGetInfo

```
void PDSysFontGetInfo (PDSysFont sysFont, PDEFontInfoRecP infoP,
    ASUns32 infoSize);
```

| | |
|------------------------|---|
| Description | Obtains high-level information about a system font. |
| Parameters | <p><i>sysFont</i></p> <p>A <i>PDSysFont</i> object referencing a system font whose information is obtained.</p> <p><i>infoP</i></p> <p>(Filled by the method) Pointer to PDEFontInfoRec structure to fill with font information for <i>sysFont</i>. No more than <i>infoSize</i> bytes are written to this buffer.</p> <p><i>infoSize</i></p> <p>Size of the <i>infoP</i> buffer, in bytes.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PSFCalls.h</i> |
| Related Methods | PDSysFontAcquirePlatformData PDSysFontGetEncoding PDSysFontGetName PDSysFontGetType0Widths |
| Availability | Plug-ins: Available if <i>PI_PDSYSFONT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSysFontGetName

[ASAtom](#) PDSysFontGetName ([PDSysFont](#) sysFont) ;

| | |
|------------------------|---|
| Description | Obtains the PostScript or TrueType styled name for a system font. |
| Parameters | <p><i>sysFont</i></p> <p>A <i>PDSysFont</i> object referencing a system font whose name is obtained.</p> |
| Read/Write | Write |
| Return Value | The <i>ASAtom</i> for the system font's name. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PSFCalls.h</i> |
| Related Methods | PDSysFontAcquirePlatformData PDSysFontGetEncoding PDSysFontGetInfo PDSysFontGetType0Widths |
| Availability | Plug-ins: Available if <i>PI_PDSYSFONT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSysFontGetType0Widths

```
void PDSysFontGetType0Widths (PDSysFont sysFont,
    ASAtom ordering, ASBool* hasDW, ASInt32* dw, CosObj* w,
    ASBool* hasdw2, ASInt32* dw2, CosObj* w2);
```

| | |
|--------------------|--|
| Description | Obtains width information from a Type 0 system font. This information can be used to create a <i>PDEFont</i> from a system font. |
| Parameters | <p><i>sysFont</i></p> <p>A <i>PDSysFont</i> object referencing a multibyte system font.</p> <p><i>ordering</i></p> <p><i>ASAtom</i> representing the CIDFont's Ordering information. Used to get a CMap object for <i>sysFont</i>.</p> <p><i>hasdw</i></p> <p><i>(Filled by the method)</i> <i>true</i> if <i>sysFont</i> has a valid <i>dw</i> value, <i>false</i> otherwise.</p> <p><i>dw</i></p> <p><i>(Filled by the method)</i> Default width for glyphs in a CIDFont. Currently, always 1000. See Section 7.7.9 on CIDFontType 0 in the Portable Document Format Reference Manual for more information.</p> <p><i>w</i></p> <p><i>(Filled by the method)</i> A Cos array of a set of lists that define the widths for the glyphs in the CIDFont. Each list can specify individual widths for consecutive CIDs, or one width for a range of CIDs. See Section 7.7.11 on character widths in CIDFonts in the Portable Document Format Reference Manual for information on the format of this array.</p> <p><i>hasdw2</i></p> <p><i>(Filled by the method)</i> <i>true</i> if <i>sysFont</i> has a valid <i>dw2</i> value. Default is <i>false</i>.</p> |

dw2

(Filled by the method) The default metrics for writing mode 1. This entry is an array of two *ASInt32* numbers: the *y* component of the position vector and the *y* component of the displacement vector for writing mode 1. The *x* component of the position vector is always half the width of the character. The *x* component of the displacement vector is always 0. The default value is [880 -1000]. For information on writing mode 1, see Section 7.7.12 on vertical writing in the [Portable Document Format Reference Manual](#).

w2

(Filled by the method) A Cos array defining the metrics for vertical writing. Its format is similar to the format of the array in *w*. It defines the *x* and *y* components of the position vector, and the *y* component of the displacement vector. The *x* component of the displacement vector is always 0. See Section 7.7.11 on character widths in CIDFonts in the [Portable Document Format Reference Manual](#) for information on the format of this array.

| | |
|------------------------|--|
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PSFCalls.h</i> |
| Related Methods | PDFontGetWidths |
| Availability | Plug-ins: Available if <i>PI_PDSYSFONT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSysFontGetWidths

```
void PDSysFontGetWidths (PDSysFont sysFont, ASInt16* widthsP);
```

Description Obtains the widths of a single byte encoded system font.

Parameters

sysFont
A *PDSysFont* object referencing a system font whose widths are obtained.

widthsP
(Filled by the method) Pointer to widths array. *widthsP* must have room for 256 entries.

Read/Write Write

Return Value None

Exceptions [peErrCantGetWidths](#)

Notifications None

Header File *PSFCalls.h*

Related Methods [PDSysFontAcquirePlatformData](#)
[PDSysFontGetEncoding](#)
[PDSysFontGetInfo](#)
[PDSysFontGetName](#)

Availability Plug-ins: Available if *PI_PDSYSFONT_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSysFontGetWidthsEx

```
void PDSysFontGetWidthsEx (PDSysFont sysFont, ASInt16* widthsP,  
    ASFixed* mmDesignVector);
```

| | |
|------------------------|--|
| Description | Obtains the widths of a single byte encoded system font. |
| Parameters | <p><i>sysFont</i></p> <p>A <i>PDSysFont</i> object referencing a system font whose widths are obtained.</p> <p><i>widthsP</i></p> <p>(Filled by the method) Pointer to widths array. <i>widthsP</i> must have room for 256 entries.</p> <p><i>mmDesignVector</i></p> <p>If <i>sysFont</i> is a multiple master font, points to the design vector, whose length must equal the number of design axes of <i>sysFont</i>.</p> |
| Read/Write | Write |
| Return Value | None |
| Exceptions | peErrCantGetWidths |
| Notifications | None |
| Header File | <i>PSFCalls.h</i> |
| Related Methods | PDSysFontAcquirePlatformData PDSysFontGetEncoding PDSysFontGetInfo PDSysFontGetName |
| Availability | Plug-ins: Available if <i>PI_PDSYSFONT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSysFontReleasePlatformData

```
void PDSysFontReleasePlatformData  
(PDSysFontPlatDataP platDataP);
```

| | |
|------------------------|--|
| Description | Releases platform-specific data for the specified <i>PDSysFont</i> . |
| Parameters | <i>platDataP</i> A pointer to a PDSysFontPlatDataP structure containing platform-specific data. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PSFCalls.h</i> |
| Related Methods | PDSysFontAcquirePlatformData |
| Availability | Plug-ins: Available if <i>PI_PDSYSFONT_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSEdit

PDSAttrObj

PDSAttrObjCreate

```
void PDSAttrObjCreate (PDDoc pdDoc, ASAtom owner,
    ASBool indirect, PDSAttrObj* attrObj);
```

| | |
|------------------------|---|
| Description | Creates a new attribute object with the specified owner. |
| Parameters | <p><i>pdDoc</i> Document in which the attribute object is created.</p> <p><i>owner</i> Owner of the new attribute object.</p> <p><i>indirect</i> If <i>true</i>, creates the attribute object as an indirect Cos object and sets <i>pdDoc</i>'s <i>PDDocNeedsSave</i> flag (see PDDocFlags). If <i>false</i>, creates the attribute object as a direct object.</p> <p><i>attrObj</i> (Filled by the method) The newly-created attribute object.</p> |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSAttrObjCreateFromStream |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSAttrObjCreateFromStream

```
void PDSAttrObjCreateFromStream (ASAtom owner,
    CosObj cosStreamObj, PDSAttrObj* attrObj);
```

| | |
|------------------------|--|
| Description | Creates a new attribute object with the specified owner from the specified Cos stream. |
| Parameters | <p><i>owner</i></p> <p>Owner of the new attribute object.</p> <p><i>cosStreamObj</i></p> <p>The Cos stream containing the data with which to create the attribute.</p> <p><i>attrObj</i></p> <p><i>(Filled by the method)</i> Pointer to the newly-created attribute object.</p> |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSAttrObjCreate |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSAttrObjGetOwner

[ASAtom](#) PDSAttrObjGetOwner ([PDSAttrObj](#) element);

| | |
|------------------------|---|
| Description | Gets the value of the key (Owner) in the specified attribute object. |
| Parameters | <p><i>element</i></p> <p>The element whose owner is obtained.</p> |
| Return Value | The <i>ASAtom</i> for the owner's name. |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSAttrObjCreate |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSCClassMap

PDSCClassMapAddAttrObj

```
void PDSCClassMapAddAttrObj (PDSCClassMap classMap,
    ASAtom classAtom, PDSAttrObj attrObj);
```

Description Adds the specified attribute object to the specified *PDSCClassMap* for the given class name. If the attribute object is already present, it is not added a second time.

Parameters *classMap*
The *PDSCClassMap* to which the specified attribute object is added.

classAtom
The *ASAtom* representing the class name.

attrObj
Attribute object to add to the class in *classAtom*.

Return Value None

Exceptions Raises [pdsErrBadPDF](#) if an error is found in the PDF file.

Notifications None

Header File *PDSWriteCalls.h*

Related Methods [PDSCClassMapGetAttrObj](#)
[PDSCClassMapRemoveAttrObj](#)

Availability Plug-ins: Available if *PI_PDS_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSCClassMapGetAttrObj

```
void PDSClassMapGetAttrObj ( PDSClassMap classMap,
    ASAtom classAtom, ASInt32 index, PDSAttrObj* attrObj);
```

Description Gets the attribute object associated with the specified class name at an index in the class.
If there is only one object and *index* is zero, that object is retrieved.

Parameters

classMap
The *PDSClassMap*.

classAtom
The *ASAtom* of a class name for which an associated attribute objects is found.

index
Index of the desired attribute object in the class.

attrObj
(Filled by the method) Attribute object at *index*.
Set to *CosNull* if there is no attribute object at the specified location.

Return Value None

Exceptions Various

Notifications None

Header File *PDSReadCalls.h*

Related Methods [PDSClassMapAddAttrObj](#)
[PDSClassMapGetNumAttrObjs](#)

Availability Plug-ins: Available if *PI_PDS_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | — | M, W, U |

PDSCClassMapGetNumAttrObjs

```
ASInt32 PDSCClassMapGetNumAttrObjs (PDSCClassMap classMap,  
    ASAtom classAtom);
```

| | |
|------------------------|--|
| Description | Gets the number of attribute objects associated with a class name. |
| Parameters | <i>classMap</i> The <i>PDSCClassMap</i> . <i>classAtom</i> The <i>ASAtom</i> of a class name for which the number of associated attribute objects is found. |
| Return Value | Number of attribute objects associated with the class in <i>classAtom</i> . |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSCClassMapGetAttrObj |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSCClassMapRemoveAttrObj

```
void PDSCClassMapRemoveAttrObj (PDSCClassMap classMap,
    ASAtom classAtom, PDSAttrObj attrObj);
```

| | |
|------------------------|--|
| Description | Removes the specified attribute object from the specified <i>PDSCClassMap</i> . If <i>classAtom</i> is <i>ASAtomNull</i> , removes all occurrences of <i>attrObj</i> in the entire <i>classMap</i> . |
| Parameters | <p><i>classMap</i></p> <p>The <i>PDSCClassMap</i> from which the specified attribute object is removed.</p> <p><i>classAtom</i></p> <p>The <i>ASAtom</i> of a class name for which the associated attribute object is found.</p> <p><i>attrObj</i></p> <p>Attribute object to remove from <i>classMap</i>.</p> |
| Return Value | None |
| Exceptions | Raises pdsErrBadPDF if an error is found in the PDF file. |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSCClassMapAddAttrObj PDSCClassMapRemoveClass |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSCClassMapRemoveClass

```
void PDSCClassMapRemoveClass (PDSCClassMap classMap,  
    ASAtom classAtom);
```

| | |
|------------------------|--|
| Description | Removes the specified class from the specified <i>PDSCClassMap</i> , if it exists. |
| Parameters | <i>classMap</i> The <i>PDSCClassMap</i> from which a class is removed. <i>classAtom</i> The <i>ASAtom</i> representing the class to remove from <i>classMap</i> . |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSCClassMapRemoveAttrObj |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElement

PDSElementAddAttrObj

```
void PDSElementAddAttrObj (PDSElement element,
    PDSAttrObj attrObj);
```

| | |
|------------------------|---|
| Description | Associates the specified attribute object with an element at the element's current revision value. |
| Parameters | <p><i>element</i></p> <p>Element with which <i>attrObj</i> is associated.</p> <p><i>attrObj</i></p> <p>Attribute object to associate with <i>element</i>.</p> |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSElementGetAttrObj PDSElementRemoveAttrObj |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementAddClass

```
void PDSElementAddClass (PDSElement element, ASAtom classAtom);
```

Description Adds a class name to the element's list of classes to which it belongs.

Parameters

element
Element to which a class is added.

classAtom
The *ASAtom* representing the class to add to *element*. If *classAtom* is already present among *element*'s classes, it will not be added again.

Return Value None

Exceptions Various

Notifications None

Header File *PDSWriteCalls.h*

Related Methods [PDSElementGetClass](#)
[PDSElementGetNumClasses](#)
[PDSElementRemoveAllClasses](#)
[PDSElementRemoveClass](#)

Availability Plug-ins: Available if *PI_PDS_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementClearID

```
void PDSElementClearID (PDSElement element);
```

| | |
|------------------------|---|
| Description | Removes an element's ID, if it exists. |
| Parameters | <i>element</i> Element whose ID is removed. |
| Return Value | None |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i> . Raises pdsErrBadPDF if an error is found in the PDF file. |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSElementGetID PDSElementSetID |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementCreate

```
void PDSElementCreate (PDDoc pdDoc, PDSElement* element);
```

| | |
|------------------------|---|
| Description | Creates a new (but empty) <i>PDSElement</i> . |
| Parameters | <p><i>pdDoc</i></p> <p>The <i>PDDoc</i> in which the <i>PDSElement</i> is created.</p> <p><i>element</i></p> <p>(Filled by the method) The newly-created <i>PDSElement</i>.</p> |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementGetAlt

```
ASInt32 PDSElementGetAlt (PDSElement element, ASUns8* buffer);
```

Description Gets the alternate text associated with an element.
Can first be called with a *NULL* buffer to find the size, so that *buffer* can then be appropriately sized.

Parameters

element

The element whose alternate text is obtained.

buffer

(Filled by the method) A buffer into which the alternate text is placed. May be *NULL*, if being called only to find the length of the element's alternate text.

Return Value Number of characters in *element*'s alternate text.

Exceptions Raises [pdsErrWrongTypeParameter](#) if *element* is not a valid *PDSElement*.

Notifications None

Header File *PDSReadCalls.h*

Related Methods [PDSElementSetAlt](#)

Availability Plug-ins: Available if *PI_PDS_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSElementGetAttrObj

```
ASInt32 PDSElementGetAttrObj (PDSElement element,
    ASInt32 index, PDSAttrObj* attrObj);
```

| | |
|------------------------|--|
| Description | Gets the attribute object at a specified array index in the specified element. If there is only one attribute object (that is, there is no array of attributes), and <i>index</i> is zero, that attribute object is obtained. |
| Parameters | <p><i>element</i></p> <p>The element whose attribute is obtained.</p> <p><i>index</i></p> <p>Index of the attribute object to obtain.</p> <p><i>attrObj</i></p> <p>(Filled by the method) Attribute object at <i>index</i>.</p> |
| Return Value | Revision number of <i>element</i> at time of last association. |
| Exceptions | <p>pdsErrRequiredMissing</p> <p>Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i>.</p> |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSElementAddAttrObj PDSElementGetNumAttrObjs PDSElementRemoveAttrObj |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSElementGetClass

```
ASInt32 PDSElementGetClass (PDSElement element, ASInt32 index,  
    ASAtom* classAtom);
```

Description Gets the class name at an array index in the specified element.
If there is only one attribute object (that is, there is no array), and *index* is zero, that class name is obtained.

Parameters

element
The element whose class is obtained.

index
Index of the class to obtain.

classAtom
(Filled by the method) The *ASAtom* describing the class.

Return Value Revision number of *element* at time of last association.

Exceptions [pdsErrRequiredMissing](#)
Raises [pdsErrWrongTypeParameter](#) if *element* is not a valid *PDSElement*.

Notifications None

Header File *PDSReadCalls.h*

Related Methods [PDSElementAddClass](#)
[PDSElementGetNumClasses](#)
[PDSElementRemoveAllClasses](#)
[PDSElementRemoveClass](#)

Availability Plug-ins: Available if *PI_PDS_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSElementGetFirstPage

```
CosObj PDSElementGetFirstPage (PDSElement element,  
    ASAtom* firstKidType, CosObj* firstCosObjKidOnAPage,  
    PDEContainer* firstMCKidOnAPage);
```

Description Gets the Cos object for the page of the first kid of the element.

Note: The order in which the returned page is first is the order of kids, not the order of pages. That is, the first descendant with page content determines which page is returned.

Parameters

element

Element whose kid's first page is found.

firstKidType

(Filled by the method) Pointer to an *ASAtom* for the name that appears as the **Type** entry of the actual first kid of *element*. Possible values are the values that [PDSElementGetKid](#) can return. Pass *NULL* to inhibit setting *firstKidType*.

firstCosObjKidOnAPage

(Filled by the method) The kid whose content determined that the page returned was the first page with content—if that kid is a *CosObj*. Pass *NULL* to inhibit setting *firstCosObjKidOnAPage*.

firstMCKidOnAPage

(Filled by the method) The kid whose content determined that the page returned was the first page with content—if that kid is marked content that is *not* a *CosObj*. Pass *NULL* to inhibit setting *firstMCKidOnAPage*.

Return Value

The *CosObj* of the page found, *CosObjNull* if the element has no page content.

Exceptions

Various

Notifications

None

Header File

PDSReadCalls.h

Related Methods

[PDSElementGetKid](#)

Example

```
cosPage = PDSElementGetFirstPage(pdsElement,  
    NULL, NULL, NULL);
```

Availability

Plug-ins: Available if *PI_PDS_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSElementGetID

```
ASInt32 PDSElementGetID (PDSElement element, ASUns8* idBuf);
```

Description Gets the ID of an element or *CosObjNull* if there is no ID set.

Parameters *element*
Element whose ID is obtained.

idBuf
(Filled by the method) Pointer to the buffer containing the element's ID.

Return Value The number of bytes in the ID, or zero if the element has no ID.

Exceptions Raises [pdsErrWrongTypeParameter](#) if *element* is not a valid *PDSElement*.
Raises [pdsErrBadPDF](#) if an error is found in the PDF file.

Notifications None

Header File *PDSReadCalls.h*

Related Methods [PDSElementClearID](#)
[PDSElementSetID](#)
[PDSTreeRootGetElementFromID](#)

Availability Plug-ins: Available if *PI_PDS_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSElementGetKid

```
ASAtom PDSElementGetKid (PDSElement element, ASInt32 index,
    CosObj* cosObjKid, void** pointerKid, CosObj* cosPage);
```

Description

Gets the kid at an array index in the specified element.

A PDF structural element—unlike the structure tree root—can have *several different kinds* of children: marked content, another element, or an entire PDF object. The parameter in which the kid is placed depends on the type of kid. If the kid is a structural element or an object reference, *PDSElementGetKid* places the result in *cosObjKid*; if the kid is page content, it is placed in *pointerKid*.

Any or all of *cosObjKid*, *pointerKid*, and *cosPage* can be *NULL* to get the kid's type without setting that parameter.

Note: When the kid is an MC, it is actually a pointer of the type PDEContainer. As with all PDFEdit objects, you must be careful to manage the reference count of the object by calling PDEAcquire and PDERelease. PDSElementGetKid does not call PDEAcquire for you.

Parameters

element

Element whose specified kid is found.

index

Index of the kid to obtain.

cosObjKid

(Filled by the method) The *CosObj* of the specified kid—if that kid is a *PDSElement* or an **OBJR**. If *cosObjKid* is *NULL*, it is not filled in, but the type of the kid is returned regardless.

Note: This CosObj can be treated as a PDSElement or a PDSObjR. Use the return type to decide which to use.

pointerKid

(Filled by the method) Pointer to the kid at *index*—if that kid is an **MC**. If *pointerKid* is *NULL*, it is not filled in, but the type of the kid is returned regardless.

cosPage

(Filled by the method) Pointer to the *CosObj* of the page containing the kid. If *cosPage* is *NULL*, it is not filled in, but the type of the kid is returned regardless.

| | |
|------------------------|---|
| Return Value | The <i>ASAtom</i> representing the kid's Type value: StructElem , MC or OBJR . MCR is never returned. |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i> . Raises pdsErrBadPDF if an error is found in the PDF file. |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSElementGetFirstPage PDSElementGetNumKids PDSElementInsertKid |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSElementGetNumAttrObjs

```
ASInt32 PDSElementGetNumAttrObjs (PDSElement element);
```

| | |
|------------------------|---|
| Description | Gets the number of attribute objects directly attached to the specified element. |
| Parameters | <p><i>element</i></p> <p>The element whose number of attributes is obtained.</p> |
| Return Value | Number of attribute objects directly attached to <i>element</i> . |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i> . |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSElementGetAttrObj |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSElementGetNumClasses

```
ASInt32 PDSElementGetNumClasses (PDSElement element);
```

| | |
|------------------------|---|
| Description | Gets the number of classes to which the specified element belongs. |
| Parameters | <p><i>element</i></p> <p>The element whose number of classes is obtained.</p> |
| Return Value | Number of classes to which <i>element</i> belongs. |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i> . |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSElementGetClass |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSElementGetNumKids

```
ASInt32 PDSElementGetNumKids ( PDSElement element );
```

| | |
|------------------------|---|
| Description | Gets the number of kids of the specified element. |
| Parameters | <i>element</i> Element whose number of kids is obtained. |
| Return Value | Number of direct kids of <i>element</i> . |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i> . |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSElementGetKid |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSElementGetParent

```
void PDSElementGetParent (PDSElement element,
    PDSElement* parent, ASBool* parentIsTreeRoot);
```

| | |
|------------------------|---|
| Description | Gets the immediate ancestor element of the specified element in the tree. If the element's <i>parent</i> is another element, <i>parent</i> is set to that parent and <i>parentIsTreeRoot</i> is set to <i>false</i> . If the element's <i>parent</i> is the structure tree root, <i>parent</i> is set to <i>CosNull</i> and <i>parentIsTreeRoot</i> is set to <i>true</i> . If <i>parentIsTreeRoot</i> is <i>NULL</i> , it is not set. |
| Parameters | <p><i>element</i> The element whose parent is obtained.</p> <p><i>parent</i> (Filled by the method) The <i>element</i>'s parent.</p> <p><i>parentIsTreeRoot</i> (Filled by the method) The <i>element</i>'s parent is the structure tree root.</p> |
| Return Value | None |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i> . |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSElementGetKid PDSElementGetStructTreeRoot PDSMCGetParent PDSOBJGetParent |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | — | M, W, U |

PDSElementGetRevision

```
ASInt32 PDSElementGetRevision (PDSElement element);
```

| | |
|------------------------|---|
| Description | Gets the revision number of an element. |
| Parameters | <i>element</i> The element whose revision is obtained. |
| Return Value | Revision number of <i>element</i> . |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i> . |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSElementIncrementRevision |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSElementGetStructTreeRoot

```
ASBool PDSElementGetStructTreeRoot (PDSElement element,  
    PDSTreeRoot* treeRoot);
```

| | |
|------------------------|---|
| Description | Gets the structure tree root of the document containing <i>element</i> . |
| Parameters | <i>element</i> Element whose title is obtained. <i>treeRoot</i> (Filled by the method) The structure tree root. |
| Return Value | <i>true</i> if the document has a structure tree root, <i>false</i> otherwise. If there is a structure tree root, sets <i>treeRoot</i> to be the structure tree root. |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i> . |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSElementSetTitle |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSElementGetTitle

```
ASInt32 PDSElementGetTitle (PDSElement element,  
    ASUns8* buffer);
```

- Description** Gets the title of the specified element, returning the number of characters in the title.
- Can first be called with a *NULL* buffer to find the title size, so that *buffer* can be appropriately sized as one greater than the title's length.
- Parameters**
- element*
- Element whose title is obtained.
- buffer*
- (Filled by the method) A buffer into which the title text is placed. May be *NULL*, in which case the number of characters in the title is returned.
- Return Value** Number of characters in *element*'s title, or zero if *element* has no title.
- Exceptions** Raises [pdsErrWrongTypeParameter](#) if *element* is not a valid *PDSElement*.
- Notifications** None
- Header File** *PDSReadCalls.h*
- Related Methods** [PDSElementSetTitle](#)
- Availability** Plug-ins: Available if *PI_PDS_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSElementGetType

```
ASAtom PDSElementGetType (PDSElement element);
```

| | |
|------------------------|---|
| Description | Gets the element's structural element type, which is the value of the entry in the element. All <i>PDSElements</i> have a Type value of StructElem . |
| Parameters | <i>element</i> The element whose structural element type is obtained. |
| Return Value | The <i>ASAtom</i> representing <i>element's</i> type. |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i> . |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSElementSetType |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSElementIncrementRevision

```
void PDSElementIncrementRevision (PDSElement element);
```

| | |
|------------------------|--|
| Description | Increments an element's revision count by one. |
| Parameters | <i>element</i> Element whose revision count is incremented. |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementInsertKid

```
void PDSElementInsertKid (PDSElement element, PDSElement kid,
    ASInt32 insertAfter);
```

| | |
|------------------------|---|
| Description | Inserts the specified kid <i>PDSElement</i> or OBJR object into the specified element after position <i>insertAfter</i> . |
| Parameters | <p><i>element</i></p> <p>Element in which the specified kid is inserted.</p> <p><i>kid</i></p> <p>The kid to insert.</p> <p><i>insertAfter</i></p> <p>Position after which the kid is inserted. If <i>element</i> currently has no kids, <i>insertAfter</i> is ignored.</p> |
| Return Value | None |
| Exceptions | pdsErrWrongTypeParameter |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSElementGetFirstPage PDSElementGetKid PDSElementInsertMCAsKid PDSElementRemoveKid PDSElementReplaceKid |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementInsertMCAsKid

```
void PDSElementInsertMCAsKid (PDSElement element,
    CosObj cosPage, PDSMC mc, ASInt32 insertAfter);
```

| | |
|------------------------|---|
| Description | <p>Inserts a reference to the specified <i>PDSMC</i> (marked content) in the specified element after position <i>insertAfter</i>.</p> <p>This method automatically creates MCR objects if needed.</p> |
| Parameters | <p><i>element</i></p> <p>Element in which the reference is inserted.</p> <p><i>cosPage</i></p> <p>The <i>CosObj</i> for the page containing the reference to insert.</p> <p><i>mc</i></p> <p>The marked content to insert.</p> <p><i>insertAfter</i></p> <p>Position after which the reference is inserted. If <i>element</i> currently has no kids, <i>insertAfter</i> is ignored.</p> |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSElementInsertKid PDSElementReplaceKidMC |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | — | — |

PDSElementInsertOBJAsKid

```
void PDSElementInsertOBJAsKid (PDSElement element,
    CosObj cosPage, CosObj obj, ASInt32 insertAfter);
```

| | |
|------------------------|---|
| Description | Inserts a reference to the specified PDF object as a kid into the specified element. |
| Parameters | <p><i>element</i></p> <p>Element in which the reference is inserted.</p> <p><i>cosPage</i></p> <p>The <i>CosObj</i> for the page containing the reference to insert.</p> <p><i>obj</i></p> <p>The <i>CosObj</i> to insert.</p> <p><i>insertAfter</i></p> <p>Position after which the reference is inserted in <i>element</i>. If <i>element</i> currently has no kids, <i>insertAfter</i> is ignored.</p> |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSElementReplaceKidOBJ |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementRemoveAllAttrObjs

```
void PDSElementRemoveAllAttrObjs (PDSElement element);
```

| | |
|------------------------|--|
| Description | Removes <i>all</i> attribute objects directly associated with the specified element. |
| Parameters | <i>element</i> Element whose attributes are removed. |
| Return Value | None |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i> . |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSElementRemoveAttrObj |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementRemoveAllClasses

```
void PDSElementRemoveAllClasses (PDSElement element);
```

| | |
|------------------------|--|
| Description | Removes <i>all</i> classes from the specified element. |
| Parameters | <i>element</i> Element whose classes are removed. |
| Return Value | None |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i> . |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSElementAddClass PDSElementGetClass PDSElementGetNumClasses PDSElementRemoveClass |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementRemoveAttrObj

```
void PDSElementRemoveAttrObj ( PDSElement element,  
    PDSAttrObj attrObj );
```

Description

Removes the specified attribute object from an element. If *element* does not have an *attrObj* attribute, this method does nothing.

Note: Calling PDSElementRemoveAttrObj while iterating over the attribute objects of an element will change the relationship between attribute object indices and attribute objects. Although it is possible to track this change in indices in a single loop, it is more straightforward to accumulate a list of attribute objects to remove during one pass over the attribute objects and to carry out the actual removals during a subsequent iteration over the accumulated list.

Parameters

element

Element whose attribute is removed.

attrObj

Attribute object to remove.

Return Value

None

Exceptions

Raises [pdsErrWrongTypeParameter](#) if *element* is not a valid *PDSElement* or *attrObj* is not a valid attribute object.

Notifications

None

Header File

PDSWriteCalls.h

Related Methods

[PDSElementAddAttrObj](#)
[PDSElementGetAttrObj](#)
[PDSElementRemoveAllAttrObjs](#)

Availability

Plug-ins: Available if *PI_PDS_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|---------|
| Adobe PDF Library | — | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementRemoveClass

```
void PDSElementRemoveClass (PDSElement element,  
    ASAtom classAtom);
```

Description

Removes the specified class name from the element's list of classes to which it belongs.

Note: Calling PDSElementRemoveClass while iterating over the classes of an element will change the relationship between class indices and classes. Although it is possible to track this change in indices in a single loop, it is more straightforward to accumulate a list of classes to remove during one pass over the classes and to carry out the actual removals during a subsequent iteration over the accumulated list.

Parameters

element

Element from which the specified class is removed.

classAtom

The *ASAtom* representing the class to remove.

Return Value

None

Exceptions

Raises [pdsErrWrongTypeParameter](#) if *element* is not a valid *PDSElement*.

Notifications

None

Header File

PDSWriteCalls.h

Related Methods

[PDSElementAddClass](#)
[PDSElementGetClass](#)
[PDSElementGetNumClasses](#)
[PDSElementRemoveAllClasses](#)

Availability

Plug-ins: Available if *PI_PDS_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementRemoveKid

```
void PDSElementRemoveKid (PDSElement element, CosObj kid);
```

| | |
|------------------------|--|
| Description | Removes the specified kid (<i>element</i> or OBJR) from an element. |
| Parameters | <p><i>element</i></p> <p>Element whose kid is removed.</p> <p><i>kid</i></p> <p>Kid to remove.</p> |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSElementGetKid PDSElementInsertKid PDSElementRemoveKidMC |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementRemoveKidMC

```
void PDSElementRemoveKidMC (PDSElement element, CosObj cosPage,  
    PDSMC mc);
```

Description Removes the specified *PDSMC* (marked content) from an element's kids, if it has any.

Parameters *element*

Element whose reference is removed.

cosPage

The *CosObj* for the page containing the reference to remove.

mc

The marked content to remove.

Return Value None

Exceptions Various

Notifications None

Header File *PDSWriteCalls.h*

Related Methods [PDSElementInsertMCAsKid](#)
[PDSElementReplaceKidMC](#)
[PDSElementRemoveKid](#)

Availability Plug-ins: Available if *PI_PDS_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementReplaceKid

```
void PDSElementReplaceKid (PDSElement element, CosObj oldKid,
    CosObj newKid);
```

Description Replaces the specified kid in the specified element.

Parameters *element*
Element whose kid is replaced.

oldKid
Kid to replace.

newKid
Kid that is replacing *oldKid*.

Return Value None

Exceptions Various

Notifications None

Header File *PDSWriteCalls.h*

Related Methods [PDSElementInsertKid](#)
[PDSElementGetFirstPage](#)
[PDSElementGetKid](#)
[PDSElementRemoveKid](#)
[PDSElementReplaceKidMC](#)

Availability Plug-ins: Available if *PI_PDS_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementReplaceKidMC

```
void PDSElementReplaceKidMC (PDSElement element,  
    CosObj oldCosPage, PDSMC oldMC, CosObj newCosPage,  
    PDSMC newMC);
```

Description Replaces the specified *PDSMC* (on *oldCosPage*) with a new *PDSMC* (on *newCosPage*) in the specified element.

Parameters

element
Element whose reference is replaced.

oldCosPage
The *CosObj* for the page holding the reference to replace.

oldMC
The marked content to replace.

newCosPage
The *CosObj* for the page holding the reference that is replacing *oldMC*.

newMC
The marked content that is replacing *oldMC*.

Return Value None

Exceptions Various

Notifications None

Header File *PDSWriteCalls.h*

Related Methods [PDSElementInsertMCAsKid](#)
[PDSElementRemoveKidMC](#)
[PDSElementReplaceKid](#)

Availability Plug-ins: Available if *PI_PDS_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|------|
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementReplaceKidOBJ

```
void PDSElementReplaceKidOBJ (PDSElement element,
    CosObj oldObj, CosObj newObj, CosObj newPage);
```

Description Replaces *oldObj* with *newObj* on the specified page in the specified *element*.

Parameters

element
Element whose object is replaced.

oldObj
Object to replace.

newObj
Object that is replacing *oldObj*.

newPage
The *CosObj* for the page holding the reference that is replacing *oldObj*.

Return Value None

Exceptions Various

Notifications None

Header File *PDSWriteCalls.h*

Related Methods [PDSElementInsertOBJAsKid](#)

Availability Plug-ins: Available if *PI_PDS_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementSetAlt

```
void PDSElementSetAlt (PDSElement element,
    const ASUns8* buffer, ASInt32 nBytes);
```

| | |
|------------------------|--|
| Description | Sets the alternate text representation of an element's contents. |
| Parameters | <p><i>element</i></p> <p>Element whose alternate text representation is set.</p> <p><i>buffer</i></p> <p>Pointer to a buffer containing a string to be made the element's alternate text representation.</p> <p><i>nBytes</i></p> <p>Number of bytes in <i>buffer</i> to use as <i>element</i>'s new alternate text representation. May be zero. Sets an ID even if the <i>buffer</i> length is zero, but such an ID looks like no ID according to PDSElementGetAlt.</p> |
| Return Value | None |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i> . |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSElementGetAlt |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementSetID

```
void PDSElementSetID (PDSElement element, const ASUns8* buffer,
    ASInt32 nBytes);
```

| | |
|------------------------|--|
| Description | Sets the ID of an element to the given Cos string. |
| Parameters | <p><i>element</i></p> <p>Element whose ID is set.</p> <p><i>buffer</i></p> <p>Pointer to a buffer containing a string to be made the element's ID.</p> <p><i>nBytes</i></p> <p>Number of bytes in <i>buffer</i> to use as <i>element</i>'s new ID. May be zero. Sets an ID even if the <i>buffer</i> length is zero, but such an ID looks like no ID according to PDSElementGetID.</p> |
| Return Value | None |
| Exceptions | <p>Raises pdsErrAlreadyExists if another element already has <i>id</i> as its ID.</p> <p>Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i>.</p> |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSElementGetID PDSElementClearID |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementSetTitle

```
void PDSElementSetTitle (PDSElement element,  
    const ASUns8* buffer, ASInt32 nBytes);
```

| | |
|------------------------|--|
| Description | Sets an element's title. |
| Parameters | <i>element</i> Element whose title is set. <i>buffer</i> Pointer to a buffer containing a string to be made the element's title. <i>nBytes</i> Number of bytes in <i>buffer</i> to use as <i>element</i> 's new title. May be zero. Sets a title even if the <i>buffer</i> length is zero, but such a title looks like no title according to PDSElementGetTitle . |
| Return Value | None |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i> . |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSElementGetTitle |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSElementSetType

```
void PDSElementSetType (PDSElement element, ASAtom type);
```

| | |
|------------------------|---|
| Description | Sets an element's type value (Subtype) to the specified type. |
| Parameters | <i>element</i> Element whose type is set. <i>type</i> The <i>ASAtom</i> representing the element's type. |
| Return Value | None |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>element</i> is not a valid <i>PDSElement</i> . |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSElementGetType |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSMC

PDSMCGetParent

```
void PDSMCGetParent (CosObj containingObj, PDSMC mc,
    PDSElement* parent);
```

| | |
|------------------------|---|
| Description | Gets the parent element of the specified marked content. |
| Parameters | <p><i>containingObj</i></p> <p>The <i>CosObj</i> containing the MC whose parent is obtained. For marked content on a page, this is the Cos object representing the page. For marked content elsewhere, this is the stream in which the marked content resides.</p> <p><i>mc</i></p> <p>The marked content whose parent is obtained.</p> <p><i>parent</i></p> <p>(Filled by the method) Parent element of <i>containingObj</i>.</p> |
| Return Value | None |
| Exceptions | Raises pdsErrBadPDF if an error is found in the PDF file. |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSElementInsertMCAsKid PDSElementRemoveKidMC |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSOBJR

PDSOBJGetParent

```
void PDSOBJGetParent ( CosObj obj, PDSElement* parent );
```

Description Gets the parent element of the specified PDF object.

Parameters *obj*

PDF object whose parent element is obtained. Must be referred to via an **OBJR** from some element (that is, it has a struct parent key), otherwise undefined.

parent

(Filled by the method) Parent element of *obj*.

Return Value None

Exceptions Various

Notifications None

Header File *PDSReadCalls.h*

Related Methods [PDSElementInsertKid](#)

Availability Plug-ins: Available if *PI_PDS_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSRoleMap

PDSRoleMapCopy

```
void PDSRoleMapCopy (PDSRoleMap srcRoleMap,
PDSTreeRoot dstTreeRoot, PDSRoleMap* dstRoleMap);
```

| | |
|------------------------|--|
| Description | Makes a copy of a <i>PDSRoleMap</i> , making it the <i>PDSRoleMap</i> of the specified <i>StructTreeRoot</i> . |
| Parameters | <p><i>srcRoleMap</i></p> <p>The <i>PDSRoleMap</i> to copy.</p> <p><i>dstTreeRoot</i></p> <p>The structure tree root in which to place <i>srcRoleMap</i>.</p> <p><i>dstRoleMap</i></p> <p>(Filled by the method) If not <i>NULL</i>, points to the new, copied <i>PDSRoleMap</i>.</p> |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSTreeRootGetRoleMap |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSRoleMapDoesMap

```
ASBool PDSRoleMapDoesMap (PDSRoleMap roleMap, ASAtom src,
ASAtom dst);
```

| | |
|------------------------|--|
| Description | Determines whether the specified <i>PDSRoleMap</i> provides any mapping path for two given element types. |
| Parameters | <p><i>roleMap</i></p> <p>The <i>PDSRoleMap</i>.</p> <p><i>src</i></p> <p>The <i>ASAtom</i> for an element type whose mapping is tested.</p> <p><i>dst</i></p> <p>The <i>ASAtom</i> for an element type.</p> <p><i>Note: This may be a standard element type.</i></p> |
| Return Value | <i>true</i> if an mapping path was found, <i>false</i> otherwise. |
| Exceptions | Raises pdsErrBadPDF if an error is found in the PDF file. |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSRoleMapMap |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSRoleMapGetDirectMap

```
ASAtom PDSRoleMapGetDirectMap (PDSRoleMap roleMap,  
    ASAtom type);
```

| | |
|------------------------|--|
| Description | Gets the type, if any, directly mapped in the specified <i>PDSRoleMap</i> for the given element type. |
| Parameters | <i>roleMap</i> The <i>PDSRoleMap</i> . <i>type</i> The <i>ASAtom</i> for an element type whose mapping is found. |
| Return Value | The <i>ASAtom</i> for the equivalent type specified in <i>roleMap</i> , or <i>ASAtomNull</i> if <i>type</i> has no mapping in <i>roleMap</i> . |
| Exceptions | Raises pdsErrBadPDF if an error is found in the PDF file. |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSRoleMapDoesMap |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSRoleMapMap

```
void PDSRoleMapMap (PDSRoleMap roleMap, ASAtom src,
ASAtom dst);
```

Description Maps an element type *src* to another element type *dst* in the specified *PDSRoleMap*.

Parameters *roleMap*

The *PDSRoleMap* in which to create a new mapping.

src

Element type to map to *dst*.

dst

Element type that *src* maps onto.

Note: This may be a standard element type, such as P.

Return Value None

Exceptions Raises [pdsErrAlreadyExists](#) if *src* is already mapped.

Notifications None

Header File *PDSWriteCalls.h*

Related Methods [PDSRoleMapDoesMap](#)

Availability Plug-ins: Available if *PI_PDS_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSRoleMapUnMapDst

```
void PDSRoleMapUnMapDst (PDSRoleMap roleMap, ASAtom dst);
```

| | |
|------------------------|--|
| Description | Makes the specified element type have no mapping. |
| Parameters | <p><i>roleMap</i></p> <p>The <i>PDSRoleMap</i> in which to unmap all element types that map onto the <i>dst</i> element type.</p> <p><i>dst</i></p> <p>Element type to which all mappings are removed. All element types that map to the <i>dst</i> element type are unmapped.</p> |
| Return Value | None |
| Exceptions | pdsErrWrongTypeParameter |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSRoleMapUnMapSrc |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSRoleMapUnMapSrc

```
void PDSRoleMapUnMapSrc (PDSRoleMap roleMap, ASAtom src,
    ASBool fixupOthers);
```

| | |
|------------------------|--|
| Description | Makes the specified element type have no mapping. |
| Parameters | <p><i>roleMap</i></p> <p>The <i>PDSRoleMap</i> in which to unmap the <i>src</i> element type.</p> <p><i>src</i></p> <p>Element type whose mapping is removed.</p> <p><i>fixupOthers</i></p> <p>If <i>true</i>, any element type that was directly mapped to <i>src</i> is mapped to whatever <i>src</i> previously mapped to. If <i>false</i>, <i>PDSRoleMapUnMapSrc</i> only unmaps <i>src</i>.</p> |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSRoleMapUnMapDst |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSTreeRoot

PDSTreeRootCreateClassMap

```
void PDSTreeRootCreateClassMap (PDSTreeRoot treeRoot,
    PDSClassMap* classMap);
```

| | |
|------------------------|---|
| Description | Creates a <i>PDSClassMap</i> in the specified tree root. Any previously existing <i>PDSClassMap</i> is unlinked. |
| Parameters | <p><i>treeRoot</i></p> <p>The structure tree root in which to create a <i>PDSClassMap</i>.</p> <p><i>classMap</i></p> <p>(Filled by the method) The newly-created <i>PDSClassMap</i>.</p> |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSTreeRootGetClassMap PDSTreeRootRemoveClassMap |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSTreeRootCreateRoleMap

```
void PDSTreeRootCreateRoleMap (PDSTreeRoot treeRoot,  
    PDSRoleMap* roleMap);
```

| | |
|------------------------|---|
| Description | Creates and sets the <i>PDSRoleMap</i> of the specified <i>StructTreeRoot</i> element. Any previously existing <i>PDSRoleMap</i> is unlinked. |
| Parameters | <i>treeRoot</i> The structure tree root in which to create a <i>PDSRoleMap</i> . <i>roleMap</i> (Filled by the method) The newly-created <i>PDSRoleMap</i> . |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSTreeRootGetRoleMap PDSTreeRootRemoveRoleMap |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSTreeRootGetClassMap

```
ASBool PDSTreeRootGetClassMap (PDSTreeRoot treeRoot,  
    PDSClassMap* classMap);
```

| | |
|------------------------|--|
| Description | Obtains the <i>PDSClassMap</i> object for the specified structure tree root. |
| Parameters | <i>treeRoot</i> The structure tree root whose <i>PDSClassMap</i> is obtained. <i>classMap</i> (Filled by the method) Pointer to a location in which to return the class map, if one exists. Set to <i>CosNull</i> if there is no class map. If a <i>NULL</i> pointer is passed, no retrieval will take place. |
| Return Value | <i>true</i> if there is a class map, <i>false</i> otherwise. |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSTreeRootCreateClassMap |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSTreeRootGetElementFromID

```
ASBool PDSTreeRootGetElementFromID (PDSTreeRoot treeRoot,  
    const char* id, ASInt32 numChars, PDSElement* element);
```

| | |
|------------------------|---|
| Description | Obtains the element associated with the given ID, if any. |
| Parameters | <i>treeRoot</i> The structure tree root in which to search for <i>id</i> . <i>id</i> Pointer to a buffer containing the ID to search for. <i>numChars</i> Number of characters in <i>id</i> . <i>element</i> (Filled by the method) The element corresponding to <i>id</i> . Undefined if no element has the specified <i>id</i> . |
| Return Value | <i>true</i> if an element for <i>id</i> found, or <i>false</i> with <i>element</i> undefined if the tree root contains no IDTree value. |
| Exceptions | Raises pdsErrWrongTypeParameter if <i>id</i> is <i>NULL</i> or <i>numChars</i> is zero or less. Raises pdsErrWrongTypeEntry if the IDTree value in <i>treeRoot</i> is not a dictionary. |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSElementGetID |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSTreeRootGetKid

```
void PDSTreeRootGetKid (PDSTreeRoot treeRoot, ASInt32 index,
PDSElement* kid);
```

Description Gets the kid at an array index in the specified structure tree root.

Parameters

treeRoot
The structure tree root whose kid is obtained.

index
Index of the kid to obtain.

kid
(Filled by the method) Pointer to the kid at *index*.

Return Value None

Exceptions Raises [pdsErrBadPDF](#) if an error is found in the PDF file.

Notifications None

Header File *PDSReadCalls.h*

Related Methods [PDSTreeRootGetNumKids](#)

Availability Plug-ins: Available if *PI_PDS_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSTreeRootGetNumKids

```
ASInt32 PDSTreeRootGetNumKids (PDSTreeRoot treeRoot);
```

| | |
|------------------------|---|
| Description | Gets the number of kids of the structure tree root. |
| Parameters | <p><i>treeRoot</i></p> <p>The structure tree root whose number of kids is obtained.</p> |
| Return Value | The number of kids of the structure tree root. |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSReadCalls.h</i> |
| Related Methods | PDSTreeRootGetKid |
| Availability | Plug-ins: Available if <i>PI_PDS_READ_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSTreeRootGetRoleMap

```
ASBool PDSTreeRootGetRoleMap ( PDSTreeRoot treeRoot,
    PDSRoleMap* roleMap);
```

Description Gets the *PDSRoleMap* object for the specified structure tree root.

Parameters *treeRoot*
The structure tree root whose *PDSRoleMap* is obtained.

roleMap
(Filled by the method) Pointer to a location in which to return the role map, if one exists. Set to *CosNull* if there is no role map. If a *NULL* pointer is passed, no retrieval will take place.

Return Value *true* if there is a role map, *false* otherwise.

Exceptions Various

Notifications None

Header File *PDSReadCalls.h*

Related Methods [PDSTreeRootCreateRoleMap](#)

Availability Plug-ins: Available if *PI_PDS_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|---------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | M, W, U |

PDSTreeRootInsertKid

```
void PDSTreeRootInsertKid (PDSTreeRoot treeRoot,
    PDSElement kid, ASInt32 insertAfter);
```

| | |
|------------------------|--|
| Description | Inserts the specified kid element after the given position as a kid of the specified structure tree root. |
| Parameters | <p><i>treeRoot</i></p> <p>The structure tree root in which a kid is inserted.</p> <p><i>kid</i></p> <p>The kid to insert.</p> <p><i>insertAfter</i></p> <p>Position after which the kid is inserted. If <i>element</i> currently has no kids, <i>insertAfter</i> is ignored.</p> |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSTreeRootRemoveKid PDSTreeRootReplaceKid |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSTreeRootRemoveClassMap

```
void PDSTreeRootRemoveClassMap (PDSTreeRoot treeRoot);
```

| | |
|------------------------|--|
| Description | Removes the <i>PDSClassMap</i> of the specified structure tree root element. Does nothing if one does not exist. |
| Parameters | <p><i>treeRoot</i></p> <p>The structure tree root whose <i>PDSClassMap</i> is removed.</p> |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSTreeRootCreateClassMap |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSTreeRootRemoveKid

```
void PDSTreeRootRemoveKid (PDSTreeRoot treeRoot,  
    PDSElement kid);
```

| | |
|------------------------|--|
| Description | Removes the specified kid element from the specified structure tree root. |
| Parameters | <i>treeRoot</i> The structure tree root whose kid is removed. <i>kid</i> The kid to remove. |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSTreeRootInsertKid PDSTreeRootReplaceKid |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSTreeRootRemoveRoleMap

```
void PDSTreeRootRemoveRoleMap (PDSTreeRoot treeRoot);
```

| | |
|------------------------|---|
| Description | Removes the <i>PDSRoleMap</i> of the specified structure tree root element. Does nothing if one does not exist. |
| Parameters | <i>treeRoot</i> The structure tree root whose <i>PDSRoleMap</i> is removed. |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSTreeRootCreateRoleMap |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

PDSTreeRootReplaceKid

```
void PDSTreeRootReplaceKid (PDSTreeRoot treeRoot,
    PDSElement oldKid, PDSElement newKid);
```

| | |
|------------------------|---|
| Description | Replaces structural element <i>oldKid</i> with element <i>newKid</i> as a kid of <i>treeRoot</i> . |
| Parameters | <p><i>treeRoot</i></p> <p>The structure tree root whose kid is replaced.</p> <p><i>oldKid</i></p> <p>The kid to replace.</p> <p><i>newKid</i></p> <p>The kid that is replacing <i>oldKid</i>.</p> |
| Return Value | None |
| Exceptions | Various |
| Notifications | None |
| Header File | <i>PDSWriteCalls.h</i> |
| Related Methods | PDSTreeRootInsertKid PDSTreeRootRemoveKid |
| Availability | Plug-ins: Available if <i>PI_PDS_WRITE_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|------|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | M, W |
| Reader | — | — |

Macintosh-specific

AVAppEnumSystemFonts

```
void AVAppEnumSystemFonts (ASUns32 flags,
    AVSystemFontEnumProc enumProc, void* clientData);
```

Description Enumerates a list of fonts that are installed in the system. These are the fonts that are available for use by Acrobat. This method does *not* enumerate fonts that have been extracted or fauxed by Acrobat.

Parameters

flags
For future expansion. Must be zero.

enumProc
Client provided callback to call for each system font.

clientData
Client data for *enumProc*.

Return Value None

Exceptions None

Notifications None

Header File *MacCalls.h*

Related Methods None

Availability Plug-ins: Available if *PI_MACINTOSH_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

Available in:

| | | |
|-------------------|-----------|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M | M |
| Reader | M | M |

AVAppHandleAppleEvent

```
void AVAppHandleAppleEvent (AppleEvent* appleEvent,
    DescType eventClass, DescType eventId,
    AppleEvent* replyEvent, OSErr* err);
```

Description

Handles Apple events that are sent to the Acrobat viewer. Plug-ins that wish to handle their own Apple events do so by using [HFTReplaceEntry](#) to replace this method. If your replacement method does not wish to handle an Apple event it receives, it must pass the Apple event along to the Acrobat viewer by using the [CALL REPLACED PROC](#) macro.

For further information, see “Application-Defined Routines” in the “Responding to Apple Events” chapter of *Inside Macintosh: Interapplication Communication*.

If a plug-in wishes to get/replace any of the four required Apple events, it must use Macintosh toolbox calls such as *AEGGetEventHandler* and *AESetEventHandler*.

Plug-ins can also support AppleScript; the Acrobat viewer contains an ‘scsz’ resource, and plug-ins can handle the *GetAETE* event to append information about scriptable events they provide.

Parameters

appleEvent

The Apple event that was sent to the Acrobat viewer.

eventClass

The event class of *appleEvent*.

eventId

The event ID of *appleEvent*.

replyEvent

An event to send back to the application that sent *appleEvent*. Add parameters to this event, if appropriate.

err

A value indicating whether or not the Apple event was handled successfully. Return the value *noErr* (defined by Apple, not in the Acrobat SDK) if the Apple event was handled successfully. See *Inside Macintosh: Interapplication Communication*. *err* is the value returned by the function *MyEventHandler* in that section. In the Acrobat core API, it is returned as a parameter rather than as a function's return value.

| | |
|------------------------|--|
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>MacCalls.h</i> |
| Related Methods | AVAppWindowHandlePlatformEvent |
| Availability | Plug-ins: Available if <i>PI_MACINTOSH_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M | M |
| Reader | M | M |

AVAppWindowHandlePlatformEvent

```
ASBool AVAppWindowHandlePlatformEvent (AVWindow win,
void* platformEvent);
```

| | |
|------------------------|--|
| Description | Passes an event to the given <i>AVWindow</i> . |
| Parameters | <p><i>win</i></p> <p>The window to which an event is given.</p> <p><i>platformEvent</i></p> <p>Pointer to an instance of the Macintosh's event data type, a pointer to an <i>EventRecord</i>. The following event types are handled:</p> <ul style="list-style-type: none"> • mouseDown • updateEvt • activateEvt • keyDown • autoKey |
| Return Value | <i>true</i> if the event was handled, <i>false</i> otherwise. |
| Exceptions | Various, depending on event handled. |
| Notifications | Various, depending on event handled. |
| Header File | <i>AVCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_ACROVIEW_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M | M |
| Reader | M | M |

AVRectToRect

```
void AVRectToRect (const AVRect* avr, Rect* rect);
```

| | |
|------------------------|---|
| Description | Converts an <i>AVRect</i> into a QuickDraw <i>rect</i> . |
| Parameters | <p><i>avr</i></p> <p>Pointer to the AVRect to convert to a <i>Rect</i>.</p> <p><i>rect</i></p> <p>(Filled by the method) The <i>Rect</i> corresponding to <i>avr</i>.</p> |
| Return Value | None |
| Exceptions | Numerous |
| Notifications | Numerous |
| Header File | <i>MacCalls.h</i> |
| Related Methods | RectToAVRect |
| Availability | Plug-ins: Available if <i>PI_MACINTOSH_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M | M |
| Reader | M | M |

AVWindowGetCursorAtPoint

[AVCursor](#) AVWindowGetCursorAtPoint ([AVWindow](#) win, ASInt32 xHit, ASInt32 yHit);

| | |
|------------------------|--|
| Description | Queries the <i>AVWindow</i> for the appropriate cursor for display at the given point (in the <i>AVWindow</i> 's device coordinate space). |
| Parameters | <p><i>win</i></p> <p>The <i>AVWindow</i> to query.</p> <p><i>xHit</i></p> <p>x-coordinate of point in <i>win</i>.</p> <p><i>yHit</i></p> <p>y-coordinate of point in <i>win</i>.</p> |
| Return Value | The <i>AVCursor</i> appropriate for the specified point in the window. |
| Exceptions | Various, depending on the method called. |
| Notifications | None |
| Header File | <i>AVCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_MACINTOSH_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020002 or higher. |

Available in:

| | | |
|-------------------|-----------|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M | M |
| Reader | M | M |

RectToAVRect

```
void RectToAVRect (const Rect* rect, AVRect* avr);
```

| | |
|------------------------|--|
| Description | Converts a Quickdraw <i>rect</i> into an <i>AVRect</i> . |
| Parameters | <p><i>rect</i></p> <p>Pointer to a <i>Rect</i> to convert to an <i>AVRect</i>.</p> <p><i>avr</i></p> <p>(Filled by the method) Pointer to the AVRect corresponding to <i>rect</i>.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>MacCalls.h</i> |
| Related Methods | AVRectToRect |
| Availability | Plug-ins: Available if <i>PI_MACINTOSH_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | M | M |
| Reader | M | M |

UNIX-specific

UNIXApp

UnixAppAddModifierCallback

```
void UnixAppAddModifierCallback (XtCallbackProc callback,  
                                XtPointer closure);
```

Description

Registers a callback to call when a KeyEvent that specifies a modifier key is dispatched. The modifier state can be queried with [AVSysGetModifiers](#) or *XQueryPointer*.

The callback will be called even though the state of the modifiers may not have actually changed. For example, if both the left and right control keys are depressed, then releasing one of them causes the callbacks to be called, even though the state has not changed.

Parameters

callback

The procedure to call. Must be declared in the form:

```
static void UnixPageViewModifierCallback  
(Widget widget, XtPointer clientData,  
 XtPointer callData)
```

where *callData* is an *eventPtr*, and *widget* is the acrobat viewer's shell widget (as returned by [UnixAppGetAppShellWidget](#)).

closure

User-supplied data to pass (as *clientData*) to *proc* each time it is called.

Return Value

None

Exceptions

[genErrNoMemory](#)

Notifications

None

Header File

UnixCalls.h

Related Methods

[UnixAppRemoveModifierCallback](#)

Availability

Plug-ins: Available if *PI_UNIX_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|-----|-----|
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|-----|
| Acrobat | U | U |
| Reader | U | U |

UnixAppClipboardGetItemId

```
long UnixAppClipboardGetItemId (Display** displayPtr,
                               Window* windowPtr);
```

| | |
|------------------------|---|
| Description | Gets the item_id, display, and window needed to call <i>XmClipboardCopy</i> during a Copy or Cut operation of a custom selection server (see AVDocSelectionServer). The Acrobat viewer will have already started the Motif Clipboard Copy or Cut operation by calling <i>XmClipboardStartCopy</i> before calling the selection server's AVDocSelectionCopyProc or AVDocSelectionCutProc function. When the Copy or Cut function returns, the Acrobat viewer calls <i>XmClipboardEndCopy</i> . |
| Parameters | <i>displayPtr</i> (Filled by the method) The clipboard's display. <i>windowPtr</i> (Filled by the method) The clipboard's window. |
| Return Value | The item_id. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>UnixCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_UNIX_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixAppDispatchEvent

```
ASBool UnixAppDispatchEvent (XEvent* eventPtr);
```

| | |
|------------------------|--|
| Description | A wrapper function for <i>XtDispatchEvent</i> . The Acrobat viewer needs to look at every event before it is dispatched by <i>libXt</i> . If a plug-in needs to dispatch events, it must dispatch them by either calling <i>UnixAppDispatchEvent</i> or UnixAppProcessEvent . |
| Parameters | <i>eventPtr</i> The event. |
| Return Value | <i>true</i> if the event was dispatched, <i>false</i> if no handler was found to dispatch the event to. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>UnixCalls.h</i> |
| Related Methods | UnixAppProcessEvent |
| Availability | Plug-ins: Available if <i>PI_UNIX_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixAppGetAppShellWidget

```
Widget UnixAppGetAppShellWidget (void);
```

| | |
|------------------------|--|
| Description | Gets the application shell widget created when the Acrobat viewer called <i>XtAppInitialize</i> . The widget can be used to get the display, screen, <i>XtAppContext</i> , and so forth. Each plug-in should create its own application shell widget, which will be the parent widget for all dialogs the plug-in creates. |
| Parameters | None |
| Return Value | The Acrobat viewer's application shell widget. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>UnixCalls.h</i> |
| Related Methods | UnixAppAddModifierCallback |

Example

```
#define PLUG_IN_CLASS "Foo"

static String fooFallbackResources[] = {
    "Foo*XmText.background:green",
    "Foo*shadowThickness:5",
    NULL
};

static Widget fooAppShellWidget;

FooInit()
{
    Widget appShellWidget =
    UnixAppGetAppShellWidget();

    ...

    UnixAppLoadPlugInAppDefaults(PLUG_IN_CLASS,
    fooFallbackResources);

    fooAppShellWidget = XtVaAppCreateShell(
    XtName(appShellWidget), PLUG_IN_CLASS,
    applicationShellWidgetClass,
    XtDisplay(appShellWidget),
    XmNmappedWhenManaged, False, XmNx,
    WidthOfScreen(XtScreen(appShellWidget)) / 2
    , XmNy,
    HeightOfScreen(XtScreen(appShellWidget)) /
    2, XmNwidth, 1, XmNheight, 1, NULL);

    XtRealizeWidget(fooAppShellWidget);

    ...
}
```

Note, setting the x, y, width and height will cause all dialogs to be centered on the screen. Also, setting *XmNmappedWhenManaged* to *false* makes it invisible (that is, unmapped).

Availability

Plug-ins: Available if *PI_UNIX_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixAppGetPlugInFilename

```
char* UnixAppGetPlugInFilename (ASExtension thePI);
```

| | |
|------------------------|---|
| Description | Gets the plug-in's file name. The directory can be used to find auxiliary files for that plug-in. The Acrobat viewer searches all the directories specified in the <i>systemPlugInPath</i> and <i>userPlugInPath</i> resources to find plug-ins. |
| Parameters | <i>thePI</i> The gExtensionID extension registering the plug-in. |
| Return Value | The plug-in's file name. The string returned must not be altered or freed. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>UnixCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_UNIX_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixAppLoadPlugInAppDefaults

```
void UnixAppLoadPlugInAppDefaults (String className,  
    String fallbackResources);
```

Description

Loads the system and user application defaults for a plug-in into the screen resource database.

The screen resource database is shared by all plug-ins, and the Acrobat viewer, so name space is important.

Application default file names are generated from the *XFILESEARCHPATH* and *XUSERFILESEARCHPATH* environment variables, and the *className* parameter.

Parameters

className

Should be the same as the class name used to create the plug-in's application shell widget.

fallbackResources

NULL-terminated array of resource specification strings to use if the system application default file is not found. See *XtAppInitialize* for more information.

fallbackResources should always start with the class name of the plug-in so they do not interfere with other plug-ins, or Acrobat's resources. That is why it is suggested to create an application shell widget for each plug-in, with a unique class name.

Return Value

None

Exceptions

[genErrNoMemory](#)

Notifications

None

Header File

UnixCalls.h

Related Methods

[UnixSysPrefInit](#)
[UnixSysPrefUpdate](#)

Example

```
#define PLUG_IN_CLASS "Foo"

static String fooFallbackResources[] = {
    "Foo*XmText.background:green",
    "Foo*shadowThickness:5",
    NULL
};

static Widget fooAppShellWidget;

FooInit()
{
    Widget appShellWidget =
    UnixAppGetAppShellWidget();

    ...

    UnixAppLoadPlugInAppDefaults(PLUG_IN_CLASS,
    fooFallbackResources);

    fooAppShellWidget = XtVaAppCreateShell(
    XtName(appShellWidget), PLUG_IN_CLASS,
    applicationShellWidgetClass,
    XtDisplay(appShellWidget),
    XmNmappedWhenManaged, False, XmNx,
    WidthOfScreen(XtScreen(appShellWidget)) / 2,
    XmNy,
    HeightOfScreen(XtScreen(appShellWidget)) /
    2, XmNwidth, 1, XmNheight, 1, NULL);

    XtRealizeWidget(fooAppShellWidget);

    ...
}
```

Availability

Plug-ins: Available if *PI_UNIX_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|-----|
| Acrobat | U | U |
| Reader | U | U |

UnixAppProcessEvent

```
void UnixAppProcessEvent (XtAppContext appContext,
    XtInputMask mask);
```

| | |
|------------------------|--|
| Description | <p>A wrapper function for <i>XtAppProcessEvent</i>.</p> <p>The Acrobat viewer needs to look at every event before it is dispatched by <i>libXt</i>. If a plug-in needs to dispatch events, it must dispatch them by either calling UnixAppDispatchEvent or <i>UnixAppProcessEvent</i>.</p> |
| Parameters | <p><i>appContext</i></p> <p>The application context that identifies the application; same as the parameter passed to <i>XtAppProcessEvent</i>.</p> <p><i>mask</i></p> <p>A mask specifying which events to handle; same as the parameter passed to <i>XtAppProcessEvent</i>.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>UnixCalls.h</i> |
| Related Methods | UnixAppDispatchEvent |
| Availability | Plug-ins: Available if <i>PI_UNIX_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixAppRemoveModifierCallback

```
void UnixAppRemoveModifierCallback (XtCallbackProc callback,
    XtPointer closure);
```

| | |
|------------------------|---|
| Description | Removes a callback added by UnixAppAddModifierCallback . Both <i>closure</i> and <i>callback</i> must match the values passed in the call to UnixAppAddModifierCallback . |
| Parameters | <p><i>callback</i></p> <p>The callback to remove.</p> <p><i>closure</i></p> <p>User-supplied data that was passed in the call to UnixAppAddModifierCallback.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>UnixCalls.h</i> |
| Related Methods | UnixAppAddModifierCallback |
| Availability | Plug-ins: Available if <i>PI_UNIX_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixAppWaitForWm

```
void UnixAppWaitForWm (Widget shellWidget);
```

| | |
|------------------------|--|
| Description | <p>Dispatches events until the shell is mapped or the timeout specified by the shell's <i>XtNwaitForWm</i> and <i>XmNwmTimeout</i> has expired.</p> <p>This method can be used to wait for the window manager to map the shell window when bringing up a dialog.</p> |
| Parameters | <p><i>shellWidget</i></p> <p>Must be a ShellWidgetClass widget, and it must be realized.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>UnixCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_UNIX_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | W | W |
| Reader | W | W |

UNIXSys

UnixSysGetConfigName

```
char* UnixSysGetConfigName (void);
```

Description

Gets the Acrobat viewer's configuration name. The name will be one of:

SunOS — *sparcsun*

Solaris — *sparcsolaris*

HP-UX — *hppahpux*

The Acrobat viewer's launch shell script uses *uname* to set the configuration name in the environment variable *ACRO_CONFIG*.

Auxiliary files can be found by concatenating the installation directory with the configuration name sub-directory:

```
"<installation_dir>/<config_name>"
```

Parameters

None

Return Value

The Acrobat viewer's configuration name.

Exceptions

None

Notifications

None

Header File

UnixCalls.h

Related Methods

None

Availability

Plug-ins: Available if *PI_UNIX_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixSysGetCursor

```
Cursor UnixSysGetCursor (Widget widget, char* resourceName,  
    char* defaultName, char* defaultBits, char* defaultMaskBits,  
    unsigned int defaultWidth, unsigned int defaultHeight,  
    int defaultHotX, int defaultHotY);
```

Description

A convenience method for getting internationalized cursors. It sets up a *XtResource* specification with the *resourceName* and *defaultName* parameters, then calls *XtGetSubresources* with the name **cursor** and class **Cursor**.

The cursor name returned by *XtGetSubresources* is used to find the cursor bitmap by calling *XmGetPixmapByDepth*, which first checks the Motif image cache. If not found in the image cache, then *XmGetPixmapByDepth* searches for an *xbm* file. If the cursor name is not an absolute file name (that is, does not start with '/'), then *XmGetPixmapByDepth* uses the environment variable *XBMLANGPATH* to find it.

If a bitmap is found, then *UnixSysGetCursor* appends **Mask** to the bitmap name to find the cursor mask bitmap, otherwise it uses the *defaultBits* and *defaultMaskBits* to create a cursor.

This method does not cache its results; a new cursor is created each time it is called.

Parameters

widget

The widget to pass to *XtGetSubresources*.

resourceName

Used to create an *XtResource* specification to locate the cursor.

defaultName

Used to create an *XtResource* specification to locate the cursor.

defaultBits

Bitmap used for the cursor if the requested cursor cannot be found. It is passed to *XCreateBitmapFromData*.

defaultMaskBits

Bitmap used as the cursor mask if the requested cursor cannot be found. It is passed to *XCreateBitmapFromData*.

defaultWidth

Cursor width if the requested cursor cannot be found. It is passed to *XCreateBitmapFromData*.

defaultHeight

Cursor width if the requested cursor cannot be found. It is passed to *XCreateBitmapFromData*.

defaultHotX

x-coordinate of the cursor's hotspot if the requested cursor cannot be found. It is passed to *XCreatePixmapCursor*.

defaultHotY

y-coordinate of the cursor's hotspot if the requested cursor cannot be found. It is passed to *XCreatePixmapCursor*.

| | |
|------------------------|---|
| Return Value | The requested cursor. |
| Exceptions | genErrNoMemory |
| Notifications | None |
| Header File | <i>UnixCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_UNIX_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixSysGetCwd

```
char* UnixSysGetCwd (void);
```

| | |
|------------------------|---|
| Description | Gets the current working directory. This method tries to eliminate automounter <i>tmp</i> directories and symbol links. Using <i>stat</i> , it checks if the environment variable <i>PWD</i> specifies the same directory returned by <i>getcwd</i> . |
| Parameters | None |
| Return Value | The current working directory. |
| Exceptions | genErrNoMemory |
| Notifications | None |
| Header File | <i>UnixCalls.h</i> |
| Related Methods | UnixSysGetCwd |
| Availability | Plug-ins: Available if <i>PI_UNIX_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixSysGetHomeDirectory

```
char* UnixSysGetHomeDirectory (void);
```

| | |
|------------------------|--|
| Description | Gets the home directory of the user running the Acrobat viewer. If the <i>HOME</i> environment variable is set, its value is returned. Otherwise the method looks in the <i>passwd</i> database. |
| Parameters | None |
| Return Value | The user's home directory. The string returned by this method must not be altered or freed. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>UnixCalls.h</i> |
| Related Methods | UnixSysGetCwd UnixSysGetHostname |
| Availability | Plug-ins: Available if <i>PI_UNIX_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixSysGetHostname

```
char* UnixSysGetHostname (void);
```

| | |
|------------------------|---|
| Description | Gets the hostname. |
| Parameters | None |
| Return Value | The hostname. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>UnixCalls.h</i> |
| Related Methods | UnixSysGetHomeDirectory |
| Availability | Plug-ins: Available if <i>PI_UNIX_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixSysGetIcon

```
Pixmap UnixSysGetIcon (Widget widget, char* resourceName,  
    char* defaultName, char* defaultBits,  
    unsigned int defaultWidth, unsigned int defaultHeight);
```

Description

A convenience method for getting internationalized icons, which are bitmaps (pixmap with depth 1). Use [UnixSysGetPixmap](#) for pixmaps with depth greater than 1.

This method sets up an *XtResource* specification with the *resourceName* and *defaultName* parameters, then calls *XtGetSubresources* with the name **icon** and class **Icon**.

The icon name returned by *XtGetSubresources* is used to find the icon by calling *XmGetPixmapByDepth*, which first checks the Motif image cache. If not found in the image cache, then *XmGetPixmapByDepth* searches for an *xbm* file. If the cursor name is not an absolute file name (that is, does not start with '/'), then *XmGetPixmapByDepth* uses the environment variable *XBMLANGPATH* to find it.

If an icon is not found, *defaultBits* is used to create an icon.

This method does not cache its results, a new icon will be created each time it is called.

Parameters

widget

Widget, as passed to *XtGetSubresources*.

resourceName

Used to create an *XtResource* specification for the cursor.

defaultName

Used to create an *XtResource* specification for the cursor.

defaultBits

Bitmap used for the icon if the requested icon is not found. Passed to *XCreateBitmapFromData*.

defaultWidth

Width used for the default icon if the requested icon is not found. Passed to *XCreateBitmapFromData*.

defaultHeight

Height used for the default icon if the requested icon is not found. Passed to *XCreateBitmapFromData*.

| | |
|------------------------|---|
| Return Value | The requested icon. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>UnixCalls.h</i> |
| Related Methods | UnixSysGetPixmap |
| Availability | Plug-ins: Available if <i>PI_UNIX_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixSysGetInstallDirectory

```
char* UnixSysGetInstallDirectory (void);
```

Description

Gets the directory in which the Acrobat viewer is installed. The launch shell script sets the installation directory in the environment variable *ACRO_INSTALL_DIR*.

Auxiliary files can be found by concatenating the installation directory with the configuration name sub-directory:

```
"<installation_dir>/<config_name>"
```

Parameters

None

Return Value

The Acrobat viewer's installation directory.

Exceptions

None

Notifications

None

Header File

UnixCalls.h

Related Methods

[UnixSysGetHomeDirectory](#)

Availability

Plug-ins: Available if *PI_UNIX_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixSysGetPixmap

```
Pixmap UnixSysGetPixmap (Widget widget, char* resourceName,
    char* defaultFilename, char** defaultXpmData,
    int defaultXpmDataCount, unsigned int* widthPtr,
    unsigned int* heightPtr);
```

Description

A convenience method for getting internationalized pixmaps of depth greater than 1. This method is intended for displaying graphics in an About box. The pixmap will be created with the same depth as the widget. Use [UnixSysGetIcon](#) for bitmaps (that is, pixmaps with depth 1).

This method sets up an *XtResource* specification with the *resourceName* and *defaultFilename* parameters, then calls *XtGetSubresources* with the name **pixmap** and class **Pixmap**.

If the pixmap file name returned by *XtGetSubresources* exists and is a valid *xbm* or *xpm* file, then a pixmap will be created. Otherwise *defaultXpmData* is used.

If a grayscale visual (or the application is in grayscale mode because it could not allocate enough colors, see color cube resources for more info), then the colormap key **m** will be used in the *xpm* file (or data), otherwise **c** will be used.

XLookupColor is used to translate the colors specified in the *xpm* colormap. Because the colormap is a valuable shared resource, and *UnixSysGetPixmap* was only intended for the graphics in an About box, *XAllocColor* is not called. Instead the closest color in the Acrobat viewer's color cube is used. Therefore, it is suggested that:

- All *xpm* files (and data) have both **c** and **m** colormap keys, for both color and grayscale modes.
- Specify all colors in hexadecimal form (#rgb) to avoid problems with the Xserver's RGB database.
- Specify only the 8 fully saturated colors (black, white, red, green, ...) #000, #fff, #f00, #0f0, #00f, #ff0, #0ff, #f0f, since only these colors (which are the corners of the color cube) can be guaranteed.

This method does not cache its results, a new pixmap will be created each time it is called.

Parameters

widget

The widget, passed to *XtGetSubresources*.

resourceName

Used to create an *XtResource* specification for the pixmap.

defaultFilename

Name of a file containing a default pixmap to use if the requested pixmap is not found. Used to create an *XtResource* specification for the pixmap.

defaultXpmData

A default pixmap used if the requested pixmap cannot be found.

defaultXpmCount

The number of strings in the *defaultXpmData* array, typically *XtNumber(defaultXpmData)*.

widthPtr

Width of the image. Used both if the requested pixmap is found, and for the default pixmap, if the requested pixmap is not found.

heightPtr

Height of the image. Used both if the requested pixmap is found, and for the default pixmap, if the requested pixmap is not found.

Return Value

The requested pixmap.

Exceptions

None

Notifications

None

Header File

UnixCalls.h

Related Methods

[UnixSysGetIcon](#)

Availability

Plug-ins: Available if *PI_UNIX_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|-----|
| Acrobat | U | U |
| Reader | U | U |

UnixSysGetString

```
char* UnixSysGetString (Widget widget, char* resourceName,
    char* defaultString);
```

Description A convenience method for getting internationalized strings. It sets up an *XtResource* specification with the *resourceName* and *defaultString* parameters, then calls *XtGetSubresources* with the name **string** and class **String**.

This method does not cache its results.

Parameters

widget

The widget, passed to *XtGetSubresources*.

resourceName

Used to create an *XtResource* specification to get the string.

defaultString

Default string, used if the requested string is not found.

Return Value

The requested string, or the default string if the requested string cannot be located.

Exceptions

None

Notifications

None

Header File

UnixCalls.h

Related Methods

None

Availability

Plug-ins: Available if *PI_UNIX_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----------|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixSysGetTempFileDirectory

```
char* UnixSysGetTempFileDirectory (void);
```

| | |
|------------------------|---|
| Description | Gets the temporary file directory specified by the user. The default is <i>"/tmp."</i> |
| Parameters | None |
| Return Value | The temporary file directory. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>UnixCalls.h</i> |
| Related Methods | UnixSysGetHomeDirectory |
| Availability | Plug-ins: Available if <i>PI_UNIX_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixSysPrefInit

```
void UnixSysPrefInit (Widget widget, String name, String class,  
    char* prefFilename, XtResourceList resources,  
    Cardinal numResources, XtPointer base, XrmDatabase* dbPtr,  
    void** resourceDataPtr);
```

Description

Reads preferences from the specified file into a resource manager database. The file format is the same as an application defaults file.

The database is then used to initialize the structure pointed to by base, which is described by resources. See *XtGetApplicationResources* for more information on the format of the resources parameter.

UnixSysPrefInit uses string resource converters to convert the string data in the preference file to the appropriate data type, as described in the resources parameter. See *XtConvertAndStore* for a list of standard converters, which are automatically registered. For other data types described in the resources parameter, a converter needs to register with *XtSetTypeConverter* prior to calling *UnixSysPrefInit*.

Furthermore, a reverse string converter needs to register prior to calling [UnixSysPrefUpdate](#). The Acrobat viewer has already registered reverse string converters for the following general data types: *XtRBoolean*, *XtRBool*, *XtRDimension*, *XtRFloat*, *XtRInt*, *XtRPosition*, *XtRShort*, *XtRUnsignedChar*.

Also the following Acrobat viewer data types have both forward and reverse string converters:

```
#define XtRBoolean16 "Boolean16" /* boolean  
    */ #define XtRAVZoomType "AVZoomType"  
#define XtRFixed "ASFixed"  
#define XtRPDColorValueRec  
    "PDColorValueRec"  
#define XtRPDPageMode "PDPageMode"
```

Finally, Motif provides the function *XmRepTypeRegister*, which will create a string converter for a list of strings to an unsigned char. Also, *XmRepTypeAddReverse* will create the reverse string converter.

Parameters

widget

The plug-in's application shell. This value is used for calling *XtConvertAndStore*.

name

The name used to create the plug-in's application shell. This value is used to match items in the database.

class

The class used to create the plug-in's application shell. This value is used to match items in the database.

prefFilename

The file from which preferences are read. If non-*NULL*, *prefFileName* must be a resource file, and its contents are merged into the database pointed to by *dbPtr*.

resources

List of resources to read from the file.

numResources

Number of resources to read from the file (that is, the number of resources specified in the *resources* parameter).

base

All string data put into the structure pointed to by *base* must not be altered or freed. It is suggested that a copy of all string data be made, as shown in the Example.

dbPtr

Pointer to a resource manager database. The database is created from the specified items in the preferences file.

Plug-ins typically do not need access to the resource manager database created from the preference file, and can pass *NULL* for this parameter.

If *dbPtr* is not *NULL*, it must point to an *XrmDatabase* variable which is initialized to *NULL*, or is a valid resource manager database with which the preference file will be combined.

resourceDataPtr

Pointer to a data buffer. Must be a valid pointer. If non-*NULL*, all resources specified in the *resources parameter* or contained in the file specified by *prefFilename* are copied into this buffer.

| | |
|------------------------|-----------------------------------|
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>UnixCalls.h</i> |
| Related Methods | UnixSysPrefUpdate |

Example

```
#define PLUG_IN_CLASS "Foo"
#define PREF_FILE ".foorc"

static Widget fooAppShellWidget;

typedef struct {
    Boolean paletteOnLeft;
    String greetingString;
    int grossNationalDebt;
} FooDataRec;

static XtResource fooResources[] = {
    { "paletteOnLeft", "PaletteOnLeft",
      XtBoolean, sizeof(Boolean),
      XtOffsetOf(FooDataRec, paletteOnLeft),
      XtRImmediate, True },
    { "greetingString", "GreetingString",
      XtRString, sizeof(String),
      XtOffsetOf(FooDataRec, greetingString),
      XtRImmediate, "Hello" },
    { "grossNationalDebt", "GrossNationalDebt",
      XtRInt, sizeof(int), XtOffsetOf(FooDataRec,
      grossNationalDebt), XtRImmediate, MAX_INT },
    };

static void* fooResourceData;
static String fooPrefFilename;
static String fooFallbackResources[] = {
    "Foo*XmText.background:green",
    "Foo*shadowThickness:5",
    NULL
};

FooDataRec fooDataRec;

FooInit()
{
    Widget appShellWidget =
    UnixAppGetAppShellWidget(); String appName =
    XtName(appShellWidget); String homeDir =
    UnixSysGetHomeDirectory();
    ...
}
```

```
UnixAppLoadPlugInAppDefaults(PLUG_IN_CLASS,
                              fooFallbackResources);

fooAppShellWidget = XtVaAppCreateShell(
    appName, PLUG_IN_CLASS,
    applicationShellWidgetClass,
    XtDisplay(appShellWidget),
    XmNmappedWhenManaged, False,
    XmNx,
    WidthOfScreen(XtScreen(appShellWidget)) / 2,
    XmNy,
    HeightOfScreen(XtScreen(appShellWidget)) /
    2, XmNwidth, 1,
    XmNheight, 1,
    NULL);

XtRealizeWidget(fooAppShellWidget);

fooPrefFilename = malloc(strlen(homeDir) + 1
+ strlen(PREF_FILE) + 1);
strcpy(fooPrefFilename, homeDir);
strcat(fooPrefFilename, "/");
strcat(fooPrefFilename, PREF_FILE);

UnixSysPrefInit(fooAppShellWidget, appName,
PLUG_IN_CLASS,
fooPrefFilename, fooResources,
XtNumber(fooResources), &fooDataRec, NULL,
&fooResourceData);

if (fooDataRec.greetingString != NULL)
{
    String tmpString =
    malloc(strlen(fooDataRec.greetingString) +
    1);

    strcpy(tmpString,
    fooDataRec.greetingString);
    fooDataRec.greetingString = tmpString;
}

...
}
```

```

FooExit()
{
...

UnixSysPrefUpdate(fooAppShellWidget,
fooPrefFilename, &fooDataRec,
fooResources, XtNumber(fooResources),
fooResourceData);

...
}

```

Availability Plug-ins: Available if *PI_UNIX_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

UnixSysPrefUpdate

```
void UnixSysPrefUpdate (Widget widget, char* prefFilename,
    XtPointer base, XtResourceList resources,
    Cardinal numResources, void* resourceData);
```

Description Updates a preference file read in by [UnixSysPrefInit](#). All the remaining parameters are the same as those passed to UnixSysPrefInit.

Data in the preference file are strings. *UnixSysPrefUpdate* uses reverse string converters (for example, int to string) to update the file. Therefore, reverse string converters need to register for all data types listed in resources prior to calling *UnixSysPrefUpdate*. The Acrobat viewer has already registered reverse string converters for the following general data types: *XtRBoolean*, *XtRBool*, *XtRDimension*, *XtRFloat*, *XtRInt*, *XtRPosition*, *XtRShort*, *XtRUnsignedChar*.

and for the following Acrobat viewer data types: *XtRBoolean16*, *XtRAVZoomType*, *XtRFixed*, *XtRPDColorValueRec*, *XtRPDPageMode*.

Parameters

widget

The plug-in's application shell. This value is used for calling XtConvertAndStore.

prefFileName

The file name containing the preferences.

base

FILL ME IN

resources

FILL ME IN

numResources

FILL ME IN

resourceData

Must be the data returned by [UnixSysPrefInit](#).

Return Value None

Exceptions None

Notifications None

Header File *UnixCalls.h*

Related Methods [UnixSysPrefInit](#)

Availability Plug-ins: Available if *PI_UNIX_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | U | U |
| Reader | U | U |

Windows-specific

WinAppEnableIdleTimer

```
ASBool WinAppEnableIdleTimer (ASBool enable);
```

| | |
|------------------------|--|
| Description | <i>(Windows only)</i> Allows a plug-in to turn the <i>AVAppIdle</i> timer on and off, which is needed when a plug-in calls another process and thus blocks Acrobat for an extended period of time. |
| Parameters | <i>enable</i> <i>true</i> to turn the timer on, <i>false</i> to turn it off. |
| Return Value | The previous state of the <i>AVAppIdle</i> timer. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>WINCALLS.H</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_WIN_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | W | W |
| Reader | W | W |

WinAppGetModalParent

HWND WinAppGetModalParent ([AVDoc](#) doc);

| | |
|------------------------|--|
| Description | <p>(<i>Windows only</i>) Gets appropriate parent for any modal dialogs created by a plug-in. This method is only useful if there is an AVDoc; it cannot be used, for example, to put up a modal dialog while a file is being opened.</p> <p>In circumstances where there is no AVDoc, use the <i>gHWND</i> provided in <i>PIMAIN.C</i>. Although this does not give perfect results in some cases, there is no real alternative. (For example, if a file is opened in an external application's window, the dialog is not hidden if the external application is hidden.)</p> |
| Parameters | <p><i>doc</i></p> <p>The AVDoc for a PDF file, if the dialog is acting on an PDF document, which is generally the case. The AVDoc must be provided so that for external documents, the viewer can parent the dialog off the external application—instead of the viewer.</p> |
| Return Value | <i>HWND</i> for modal dialogs' parent. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>WINCALLS.H</i> |
| Related Methods | AVAppBeginModal AVAppEndModal |
| Availability | Plug-ins: Available if <i>PI_WIN_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | W | W |
| Reader | W | W |

WinAppGetPalette

```
HPALETTE WinAppGetPalette (void);
```

| | |
|------------------------|---|
| Description | (<i>Windows only</i>) Obtains the application's color palette in the case where the system is running in 256 color mode or less. Used when you want to set and realize a palette in an external window before drawing to it. Do <i>not</i> release this palette handle—it may be in use by other plug-ins. |
| Parameters | None |
| Return Value | The application's color palette. <i>NULL</i> if the system is running direct colors (15/16/24/32-bit) or no palette is being used. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>WINCALLS.H</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_WIN_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | W | W |
| Reader | W | W |

WinAppGetPrinterHDC

```
ACCB1 HDC ACCB2 WinAppGetPrinterHDC (void);
```

| | |
|------------------------|--|
| Description | Gets the device context for a printer, which is the <i>HDC</i> used to print a document. |
| Parameters | None |
| Return Value | The printer device context. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>WinCalls.h</i> |
| Related Methods | None |
| Availability | Plug-ins: Available if <i>PI_WIN_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00040000 or higher. |

Available in:

| | | |
|-------------------|-----------|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | W |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | W |
| Reader | — | W |

WinAppRegisterModelessDialog

```
void WinAppRegisterModelessDialog (HWND dialog);
```

| | |
|------------------------|--|
| Description | <i>(Windows only)</i> Registers modeless dialogs with the viewer so that the dialog gets the correct messages. |
| Parameters | <i>dialog</i> <i>HWND</i> for the dialog to register. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>WINCALLS.H</i> |
| Related Methods | WinAppUnRegisterModelessDialog |
| Availability | Plug-ins: Available if <i>PI_WIN_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | W | W |
| Reader | W | W |

WinAppUnRegisterModelessDialog

```
void WinAppUnRegisterModelessDialog (HWND dialog);
```

| | |
|------------------------|--|
| Description | <i>(Windows only)</i> Unregisters modeless dialogs with the viewer. |
| Parameters | <i>dialog</i> <i>HWND</i> for the dialog to unregister. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>WINCALLS.H</i> |
| Related Methods | WinAppRegisterModelessDialog |
| Availability | Plug-ins: Available if <i>PI_WIN_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|-----|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | — |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | W | W |
| Reader | W | W |

PDF Library-specific

ASAtomGetCount

```
ASInt32 ASAtomGetCount (void);
```

| | |
|------------------------|---|
| Description | Gets the number of <i>ASAtoms</i> that have been allocated. The maximum number of <i>ASAtoms</i> is 0xFFFF or 65535. <i>ASAtoms</i> cannot be deleted or freed. This method can be used to determine if it is necessary to reinitialize the library before creating more <i>ASAtoms</i> . |
| Parameters | None |
| Return Value | Number of <i>ASAtoms</i> currently allocated. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PDFLCalls.h</i> |
| Related Methods | ASAtomFromString PDFLInit PDFLTerm |

Available in:

| | | |
|-------------------|-----|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

ASPurgeMemory

```
ASSize_t ASPurgeMemory (ASSize_t amount);
```

Description Attempts to free memory from the PDF Library caches. The caches in the PDF Library can grow in complex and unexpected ways. A client may wish to manage memory use via the [PDFLInit](#) memory callbacks, or by explicitly calling this function after certain functions, such as, [PDDocClose](#).

To manage memory use via the memory callbacks, a client may call *ASPurgeMemory* during the alloc callback if it is low on memory, or if the client wishes to limit the amount of memory used by the library. This approach should be used with extreme caution and extensive testing. The run-time memory requirements are very document-specific.

Parameters *amount*
The desired amount of memory to free.

Return Value The approximate amount of memory freed.

Exceptions None

Notifications None

Header File *PDFLCalls.h*

Related Methods [PDFLInit](#)
[PDFLTerm](#)

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

AVExtensionMgrRegisterNotification

```
void AVExtensionMgrRegisterNotification (NSElector nsel,  
    ASExtension owner, void* proc, void* clientData);
```

Description Registers a user-supplied procedure to call when the specified event occurs. This is essentially identical to the [AVAppRegisterNotification](#) method. All of the PD level notifications are available with the Adobe PDF Library.

Parameters

nsel

Notification type. Must be one of the notification selectors. The notification selector is the name of the notification with the characters *NSEL* appended. For example, the selector for [PDDocDidPrintPage](#) is *PDDocDidPrintPageNSEL*.

Only the PD level notifications are available with the Adobe PDF Library.

owner

Identifies the owner.

For the Adobe PDF Library, if there is only one owner of the PDFEdit subsystem, *owner* should be zero. If there are multiple owners, each should specify a nonzero, non-negative *owner*. (A negative *owner* is reserved for the implementation.)

proc

User-supplied callback to call when the notification occurs. Its declaration depends on the notification type (see [Notifications](#)).

clientData

Pointer to user-supplied data to pass to *proc* each time it is called.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVExtensionMgrUnregisterNotification](#)

Example

```
void myWillPrintDoc(PDDoc doc, ASStm stm,
    ASInt32 level, void* clientData);
ASCallback myWillPrintDocCB =
    ASCallbackCreateNotification
        (PDDocWillPrintDoc, myWillPrintDoc);

AVExtensionMgrRegisterNotification
    (PDDocWillPrintDocNSEL, (ASExtension)1,
    myWillPrintDocCB, NULL);
```

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|-----|
| Acrobat | — | — |
| Reader | — | — |

AVExtensionMgrUnregisterNotification

```
void AVExtensionMgrUnregisterNotification (NSSelector nsel,  
    ASExtension owner, void* proc, void* clientData);
```

Description Unregisters a user-supplied procedure to call when the specified event occurs. This is essentially identical to the [AVAppUnregisterNotification](#) method.

Parameters

nsel

Notification type. Must be one of the notification selectors. The notification selector is the name of the notification with the characters *NSEL* appended. For example, the selector for [PDDocDidPrintPage](#) is *PDDocDidPrintPageNSEL*.

owner

Identifies the owner.

For the Adobe PDF Library, if there is only one owner of the PDFEdit subsystem, *owner* should be zero. If there are multiple owners, each should specify a nonzero, non-negative *owner*. (A negative *owner* is reserved for the implementation.)

proc

User-supplied callback to call when the notification occurs. Its declaration depends on the notification type (see [Notifications](#)).

You must use the *same* callback that you called [AVExtensionMgrRegisterNotification](#) with.

clientData

Pointer to user-supplied data to pass to *proc* each time it is called.

Return Value None

Exceptions None

Notifications None

Header File *AVCalls.h*

Related Methods [AVExtensionMgrRegisterNotification](#)

Example

```
AVExtensionMgrUnregisterNotification  
    (PDDocWillPrintDocNSEL, (ASExtension)1,  
    myWillPrintDocCB, NULL);
```

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

PDFLGetVersion

```
ASUns32 PDLGetVersion (void);
```

Description Obtains the value of the Adobe PDF Library version—*kPDFLVersion*. The most significant 16 bits are the major version number; the least significant 16 bits are the minor version number. The major version number indicates whether any incompatible API changes have been made. The minor version number indicates the API has changed, but in a compatible fashion.

Parameters None

Return Value Adobe PDF Library version—*kPDFLVersion*.

Exceptions None

Notifications None

Header File *PDFInit.h*

Related Methods [PDFLTerm](#)

Example

```
if(((PDFLGetVersion() & 0xffff0000) ==
    (kPDFLVersion & 0xffff0000)) )
{
    /* Major version number unchanged */
    ...
}
else
{
    /* Major version number changed! */
    ...
}
```

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|-----|
| Acrobat | — | — |
| Reader | — | — |

PDFLInit

```
ASInt32 PDFLInit (PDFLData data);
```

Description Initializes the Adobe PDF Library. This method must be called before any other Library calls can be made, printing or otherwise.

Parameters *data*
Initialization data for Adobe PDF Library.

Return Value 0 if initialization was successful, or an error code if it was not. Use [ASGetErrorString](#) to convert any error code to a string.

Exceptions None

Notifications None

Header File *PDFInit.h*

Related Methods [PDFLTerm](#)

Example

```
rtn = PDFLInit(&pdfLibData);
if (rtn) /* if not 0 then error */
{
    fprintf(stderr, "Initialization error,
        please see \"AcroErr.h\"\\n\\ for more
        information on this specific error.\\n\\n\\
Error System: %d\\nError Severity: %d\\nError
Code: %d",
        ErrGetSystem(rtn),ErrGetSeverity(rtn),ErrGet
        Code(rtn));

    exit(-1);
}
```

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|-----|
| Acrobat | — | — |
| Reader | — | — |

PDFLPrintDoc

```
void PDFLPrintDoc (PDDoc doc,  
    PDFLPrintUserParamsRec userParams);
```

Description Prints a PDF document or pages from a PDF document allowing the caller to specify options such as page size, rotation, and shrinkToFit.

Parameters

doc
The *PDDoc* for the document to print.

userParams
Parameters to control printing.

Return Value None

Exceptions [cosErrWriteError](#)
[pdErrOpNotPermitted](#)
[genErrBadParm](#)

Notifications [PSPrintAfterBeginProlog](#)
[PSPrintAfterBeginSetup](#)
[PSPrintBeforeEndSetup](#)
[PSPrintAfterBeginPageSetup](#)
[PSPrintAfterPageTrailer](#)
[PSPrintAfterTrailer](#)
[PSPrintBeforeEndComments](#)
[PSPrintAfterEmitExtGState](#)

Header File *PDFLPrint.h*

Related Methods [PDFLPrintPDF](#)

Example `PDFLPrintDoc(doc, userParams);`

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | — | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

PDFLPrintPDF

```
void PDFLPrintPDF (PDDoc pdDoc, ASPathName pathName,
    PDPrintParams psParams);
```

Description Print a document to PostScript using the specified parameters.

Parameters

pdDoc
Document to print.

pathName
Pathname to which PostScript file is printed.

psParams
Print parameters.

Return Value None

Exceptions None

Notifications None

Header File *PrintLib.h*

Related Methods None

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

PDFLTerm

```
void PDFLTerm (void);
```

Description Terminates the Adobe PDF Library. Call this method after you are completely done using the Library. Call this once to terminate and release memory used by the Library. After the Library has been shut down, the process should terminate.

Parameters None

Return Value None

Exceptions None

Notifications None

Header File *PDFInit.h*

Related Methods [PDFLInit](#)

Example `PDFToolkitTerm();`

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

Placed PDF-specific

AGMCleanup

```
void AGMCleanup (void);
```

Description Terminates AGM. You must call this method after you are completely done using AGM.

Do not call *AGMCleanup* if [AGMInit](#) returned *false*.

Parameters None

Return Value None

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [AGMInit](#)

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|-----|
| Acrobat | — | — |
| Reader | — | — |

AGMClip

```
void AGMClip (AGMPortPtr port);
```

| | |
|------------------------|--|
| Description | Sets the specified port's clip path to be the intersection of the port's current path and the current clip path. |
| Parameters | <p><i>port</i></p> <p>The port whose clip path is updated.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PXCalls.h</i> |
| Related Methods | AGMNewPath AGMMoveTo AGMLineTo AGMClosePath |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

AGMClosePath

```
void AGMClosePath (AGMPortPtr port);
```

| | |
|------------------------|--|
| Description | Closes the current path in the specified port by connecting the path's first point to its current end point. |
| Parameters | <i>port</i> The port whose current path is closed. |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PXCalls.h</i> |
| Related Methods | AGMNewPath AGMMoveTo AGMLineTo AGMFill AGMClip |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

AGMConcat

```
void AGMConcat (AGMPortPtr port, AGMFixedMatrix\* matrix,
    Bool32 isFloatMatrix);
```

Description Concatenates a specified transformation matrix onto the Current Transformation Matrix (CTM) of a specified *AGMPort*. This method can be used to rotate or skew the display of a PDF file in a port.

Parameters *port*

The port to which *matrix* is applied.

matrix

The matrix to concatenate onto *port*'s CTM. It has the form $[a \ b \ c \ d \ e \ f]$. The following matrices are especially useful:

Translate — To translate the display by a specified amount in the *x*- and *y*-directions, use a matrix of the form $[fixedOne \ 0 \ 0 \ fixedOne \ x \ y]$.

Scale — To scale the display by a specified value, use a matrix of the form $[scale \ 0 \ 0 \ scale \ 0 \ 0]$. For example, to magnify an image by a factor of 2, use *fixedTwo* as the value of *scale*.

Rotate — To rotate the display by an angle θ , use a matrix of the form $[\cos\theta \ \sin\theta \ -\sin\theta \ \cos\theta \ 0 \ 0]$.

Note: When combining several matrix operations, the order is important because matrix multiplication is non-commutative. See Sections 3.9 and 3.10 in the [Portable Document Format Reference Manual](#) for further information about transformation order.

isFloatMatrix

If *false*, *matrix* consists of fixed point numbers. If *true*, *matrix* consists of floating point numbers.

Return Value None

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [PDPPageGetFlippedMatrix](#)

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

AGMDeletePort

```
void AGMDeletePort (AGMPortPtr port);
```

Description Deletes an *AGMPort*, as well as the memory used to associate the port with either a bitmap or a window. Call this method when you are done using a bitmap or window and want to destroy it. The caller is responsible for deleting the bitmap or window itself.

Parameters *port*
The port to delete.

Return Value None

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [AGMNewBitmapPort](#)
[AGMNewRasterPort](#)
[AGMNewWindowPort](#)

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

AGMDeleteRasterDev

```
void AGMDeleteRasterDev (AGMRasterDevice* rasDev);
```

Description Destroys a specified raster device.

Parameters *rasDev*
The raster device to destroy.

Return Value None

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [AGMNewRasterDev](#)

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

AGMFill

```
void AGMFill (AGMPortPtr port);
```

Description Fills the current path in the specified port, using the color most recently set using [AGMSetRGBColor](#). The path can be constructed using [AGMNewPath](#), [AGMLineTo](#), [AGMMoveTo](#), and [AGMClosePath](#).

Parameters *port*
The port whose current path is filled.

Return Value None

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [AGMClip](#)
[AGMClosePath](#)
[AGMLineTo](#)
[AGMMoveTo](#)
[AGMNewPath](#)
[AGMSetRGBColor](#)

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|-----|
| Acrobat | — | — |
| Reader | — | — |

AGMInit

```
Bool8 AGMInit (void);
```

Description Initializes AGM. This method must be called before any other AGM methods.

When you are completely done using AGM, call [AGMCleanup](#) to terminate it.

Parameters None

Return Value *true* if initialization was successful, *false* otherwise.

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [AGMCleanup](#)

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

AGMLineTo

```
void AGMLineTo (AGMPortPtr port, AGMFixedPoint\* pt);
```

| | |
|------------------------|---|
| Description | Adds a line segment to the current path in the specified port. The segment begins at the path's previous endpoint and ends at the specified point. |
| Parameters | <p><i>port</i></p> <p>The port to whose current path the line segment is added.</p> <p><i>pt</i></p> <p>The end point of the line segment to add.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PXCalls.h</i> |
| Related Methods | AGMClip AGMClosePath AGMFill AGMMoveTo AGMNewPath AGMSetRGBColor |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

AGMMoveTo

```
void AGMMoveTo (AGMPortPtr port, AGMFixedPoint\* pt);
```

Description Sets the endpoint point of a port's current path. Does not draw a line segment between the previous endpoint and the specified point.

Parameters

port
The port whose current path endpoint is moved.

pt
The point to move to.

Return Value None

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [AGMClip](#)
[AGMClosePath](#)
[AGMFill](#)
[AGMLineTo](#)
[AGMNewPath](#)
[AGMSetRGBColor](#)

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

AGMNewBitmapPort

```
AGMPortPtr AGMNewBitmapPort (PlatformBitmapPtr bitmap,
    AGMMemObj* mem, ASInt32 flags);
```

Description

Creates a new AGM port for a bitmap device.

After creating a platform specific bitmap, call this method to add AGM drawing capability for that bitmap. This method allocates a new *AGMPort* (by calling [AGMNewRasterPort](#)) and internally associates the given bitmap with the new *AGMPort*. The new port returned may be passed to any of the Library methods. The given bitmap must remain valid as long as the *AGMPort* is in use.

Call [AGMDeletePort](#) to dispose of the port. This method deletes the port as well as the memory used to associate the port and bitmap. The caller is responsible for deleting the bitmap itself.

On the Mac OS, the *PlatformBitmapPtr* is a *GWorldPtr*. A *GrafPtr* or *CGrafPtr* also works (with type casting). If it is a real *GWorldPtr* that came from *NewGWorld*, you must call *LockPixels* on the *GWorld*'s *portPixMap* and leave the pixels locked as long as the *AGMPort* is in use. This is because AGM has no inherent ability to deal with Macintosh style unlocked, double dereferenced pointers, so the code continues to draw into the address that the pixel buffer occupied at the time the port was constructed.

In Windows, the *PlatformBitmapPtr* is an *HDC* (handle to a device context).

Parameters

bitmap

Bitmap for which a new port is created.

mem

A structure containing pointers to functions that are used by the port to allocate and free memory when needed.

flags

Flags that specify the port's characteristics.

Must be set to 0.

Return Value

Non-*NULL* if the port was successfully created, *NULL* otherwise.

| | |
|------------------------|--|
| Exceptions | None |
| Notifications | None |
| Header File | <i>PXCalls.h</i> |
| Related Methods | AGMDeletePort AGMInit AGMNewRasterPort AGMNewWindowPort PDPageDrawContents |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

AGMNewPath

```
void AGMNewPath (AGMPortPtr port);
```

Description Creates a new empty path in a specified port. The path can then be added to using [AGMMoveTo](#), [AGMLineTo](#), and [AGMClosePath](#). The resulting path can either be filled (using [AGMFill](#)) or intersected with the port's current clip path (using [AGMClip](#)).

Parameters *port*
The port in which the new path is created.

Return Value None

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [AGMClip](#)
[AGMClosePath](#)
[AGMFill](#)
[AGMLineTo](#)
[AGMMoveTo](#)
[AGMNewPath](#)
[AGMSetRGBColor](#)

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|-----|
| Acrobat | — | — |
| Reader | — | — |

AGMNewRasterDev

```
AGMRasterDevice* AGMNewRasterDev (AGMMemObj* mem,
    AGMImageAlphaRecord* image, Bool32 printerHalftoning);
```

Description Creates a new raster device that is subsequently used to create an AGM raster port. Use this method only if you wish to create an AGM port that is a raster port. In that case, you must call this method before calling [AGMNewRasterPort](#).

Parameters

mem

A structure containing pointers to functions that are used by the port to allocate and free memory when needed.

image

Pointer to a structure containing information about the alpha values for each pixel in the image.

Note: image must have its rowBytes be a multiple of 32-bits. Here is a snippet of code that calculates rowBytes such that it is rounded up to the closest 32-bits:

```
rowBytes = (((width * bitsPerPixel) + 31) >>
    5) << 2;
```

printerHalftoning

If *true*, generates a halftone pattern optimized for monochrome printers.

Return Value The newly-created raster device.

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [AGMDeleteRasterDev](#)
[AGMNewRasterPort](#)

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|-----|
| Acrobat | — | — |
| Reader | — | — |

AGMNewRasterPort

```
AGMPortPtr AGMNewRasterPort (AGMMemObj* mem, ASInt32 flags,
    AGMPortDestructProcPtr destruct, AGMRasterDevice* rasDev);
```

Description Creates a new AGM port for a raster device. The raster device must have previously been created using [AGMNewRasterDev](#).

Parameters

mem
A structure containing pointers to functions that are used by the port to allocate and free memory when needed.

flags
Flags that control the port's characteristics. Currently only one flag is defined:
kAGMNoThreads — Disables AGM multithreading for the port. If the version of AGM in use supports multithreaded ports, ports created by *AGMNewRasterPort* are multithreaded unless this flag is specified.

destruct
A procedure to call when the port is destroyed. Typically set to *NULL*.

rasDev
The raster device from which the port is created. The raster device must have previously been created using [AGMNewRasterDev](#).

Return Value The newly-created *AGMPort*.

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [AGMNewBitmapPort](#)
[AGMNewWindowPort](#)

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|-----|
| Acrobat | — | — |
| Reader | — | — |

AGMNewWindowPort

```
AGMPortPtr AGMNewWindowPort (PlatformWindowPtr window,
    AGMMemObj* mem, ASInt32 flags);
```

Description

Creates a new *AGMPort* in a specified platform-specific window to add drawing capability for that window. The window must have previously been created using platform-dependent calls. *window* must remain valid as long as the *AGMPort* is in use.

When you are finished using the window and wish to destroy it, first call [AGMDeletePort](#) to dispose of the port, then destroy the window yourself.

Parameters

window

The platform-specific window for which a new port is created:

Mac OS — *WindowPtr*

Windows — *HWND*

The window must remain valid as long as the port is in use.

mem

A structure containing pointers to functions that are used by the port to allocate and free memory when needed.

flags

Must be set to *kAGMPainPort*. *kAGMPainPort* produces an *AGMPort* that draws into (and is clipped by) *window*. It has no other side effects.

Return Value

Non-*NULL* if the port was successfully created, *NULL* otherwise.

Exceptions

None

Notifications

None

Header File

PXCalls.h

Related Methods

[AGMDeletePort](#)
[AGMInit](#)
[AGMNewBitmapPort](#)
[AGMNewRasterPort](#)
[AGMSetAntiAliasPolicy](#)

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

AGMSetAntiAliasPolicy

```
void AGMSetAntiAliasPolicy (AGMPort* port, ASInt32* policy,
    ASInt32* gridSize);
```

| | |
|--------------------|--|
| Description | Sets the anti-aliasing policy used by the specified <i>AGMPort</i> . It also returns the current policy in the same parameters. |
| Parameters | <p><i>port</i></p> <p>The <i>AGMPort</i> whose anti-aliasing policy is set.</p> <p><i>policy</i></p> <p>The anti-aliasing policy to use for this port. Must be one of the following:</p> <p><i>kAGMAANoAntiAliasing</i> — No anti-aliasing is performed. This is the default behavior. The value passed in <i>gridSize</i> is ignored for this policy.</p> <p><i>kAGMAAMultiBitMasks</i> — AGM performs the anti-aliasing by creating multi-bit text masks. In these masks, partial pixel coverage information is represented as partial transparency. AGM only attempts this kind of anti-aliasing if at least one of the monitors or bitmaps that the port is drawing on supports multiple bits per pixel and if all of the raster devices associated with those ports have the <i>kSupportsType1Runs</i> bit set in their <i>flags</i> field (this bit is usually set). The value passed in <i>gridSize</i> is ignored for this policy.</p> <p><i>kAGMAAPostDownsample</i> — Your software (not AGM) performs anti-aliasing after the entire page has been drawn. The anti-aliasing is performed by averaging the colors in a box of dimension <i>gridSize</i> × <i>gridSize</i> pixels into one pixel. The text character bitmaps that are drawn in this case (which are single bit masks) are adjusted so that when the downsampling occurs, the characters are still legible.</p> <p><i>gridSize</i></p> <p>Length (in pixels) of a cell that is averaged to perform anti-aliasing. Normal values for <i>gridSize</i> are 2 or 4.</p> <p>This is used for some values of <i>policy</i> and ignored for others. See also <i>policy</i>.</p> |

| | |
|------------------------|------------------------------------|
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PXCalls.h</i> |
| Related Methods | PDPageDrawContents |
| Available in: | |

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

AGMSetRGBColor

```
void AGMSetRGBColor (AGMPortPtr port, ASFixed red,
    ASFixed green, ASFixed blue);
```

Description

Sets the color that is used to fill paths.

This method sets the current color space to DeviceRGB and the current color value to that specified by the *red*, *green*, and *blue* values.

Parameters

port

The port whose current drawing color is set.

red

The red component of the port's new drawing color. Must be a fixed point number between *fixedZero* and *fixedOne*, inclusive.

green

The green component of the port's new drawing color. Must be a fixed point number between *fixedZero* and *fixedOne*, inclusive.

blue

The blue component of the port's new drawing color. Must be a fixed point number between *fixedZero* and *fixedOne*, inclusive.

Return Value

None

Exceptions

None

Notifications

None

Header File

PXCalls.h

Related Methods

[AGMFill](#)

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

CCSetSystemCalibration

```
ASBool CCSetSystemCalibration (void);
```

Description Obtains the display characteristics of the main monitor and passes them to the rasterizer. This allows the correct display of PDF data specified in device-independent color spaces appropriate to the current monitor.

The main monitor's characteristics are obtained using system services. In Mac OS, this method uses ColorSync. In Windows, it uses the *GetColorSpace* and *GetLogColorSpace* functions.

Parameters None

Return Value *true* if succeeded in obtaining calibration information, *false* otherwise.

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods None

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

PDDocPrintPages

```
void PDDocPrintPages (PDPrintClient client);
```

Description Prints pages from a document, controlled by a structure of data and callbacks. This method should be avoided; use [PDFLPrintDoc](#) instead.

Parameters *client*
Control structure for *PDDocPrintPages*.

Return Value None

Exceptions [genErrBadParm](#)

Notifications [PSPrintAfterBeginPageSetup](#)
[PSPrintAfterBeginProlog](#)
[PSPrintAfterBeginSetup](#)
[PSPrintAfterEmitExtGState](#)
[PSPrintAfterPageTrailer](#)
[PSPrintAfterTrailer](#)
[PSPrintBeforeEndComments](#)
[PSPrintBeforeEndSetup](#)

Header File *PDPrint.h*

Related Methods [PDFLInit](#)
[PDFLPrintDoc](#)

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|-----|
| Acrobat | — | — |
| Reader | — | — |

PDFontDownloadContextCreate

```
PDFontDownloadContext PDFontDownloadContextCreate  
(PDPrintClient client);
```

Description Creates a font download context object. This object keeps track of the fonts downloaded during a print job and whether or not substitution fonts have already been downloaded. Also tracks the font download parameters, such as *binaryOK* and emit TrueType as Type 42. For use in the [PDFontStreamPS](#) method.

Parameters *client*
The client record to pass to [PDDocPrintPages](#).

Return Value The newly-created context.

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [PDDocPrintPages](#)
[PDFontDownloadContextDestroy](#)
[PDFontStreamPS](#)

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

PDFontDownloadContextDestroy

```
void PDFontDownloadContextDestroy
(PDFontDownloadContext context);
```

Description Destroys a font download context object. Call this method after [PDDocPrintPages](#) returns.

Parameters *context*
The context to destroy.

Return Value None

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [PDDocPrintPages](#)
[PDFontDownloadContextCreate](#)

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

PDFFontStreamPS

```
ASBool PDFFontStreamPS (PDFFont fontP, ASStm stm,
    PDFFontDownloadContext context);
```

Description Emits a font into a specified stream. The font is in a format suitable for downloading to a PostScript VM. For example, a TrueType font is converted into a Type 1 or Type 42 font.

This method is only useful in the *EmitFont* callback for [PDDocPrintPages](#).

Parameters

fontP
The font to emit.

stm
The *ASStm* into which the font is emitted.

context
A context created by [PDFFontDownloadContextCreate](#).

Return Value *true* if successful, *false* otherwise.

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [PDDocPrintPages](#)
[PDFFontDownloadContextCreate](#)

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

PDFontWasExtracted

```
ASBool PDFontWasExtracted (PDFont fontP);
```

| | |
|------------------------|--|
| Description | Tests whether or not the specified font is embedded in the PDF file and has already been extracted to display or print the file. |
| Parameters | <i>fontP</i> The font to test. |
| Return Value | <i>true</i> if the font is embedded in the PDF file and has been extracted, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PXCalls.h</i> |
| Related Methods | PDFontDownloadContextCreate PDFontWasFauxed |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

PDFontWasFauxed

```
ASBool PDFontWasFauxed (PDFont font);
```

| | |
|------------------------|--|
| Description | <p>Tests whether or not the specified font</p> <p>a) is embedded in the PDF file or is installed in the user's system. In this case, the correct font can be used for display and printing.</p> <p>or</p> <p>b) is not in the PDF file and is not installed on the user's system. In this case, the Acrobat viewer has used a multiple master font to create a substitute font (also called a <i>faux</i> font).</p> |
| Parameters | <p><i>fontP</i></p> <p>The font to test.</p> |
| Return Value | <i>true</i> if the font has been substituted, <i>false</i> otherwise. |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PXCalls.h</i> |
| Related Methods | PDFontWasExtracted |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

PDFFreeMemory

```
ASSize_t PDFFreeMemory (ASSize_t amount);
```

Description Causes both the Placed PDF Library and CoolType to release any unused memory or cached data.

A client may wish to manage memory by explicitly calling *PDFFreeMemory* after certain methods, such as [PDDocClose](#).

Parameters *amount*

The desired number of bytes to free.

Return Value Approximate number of bytes freed.

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [ASAtomGetCount](#)
[PDDocClose](#)

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|-----|
| Acrobat | — | — |
| Reader | — | — |

PDPPageDrawContents

```
void PDPPageDrawContents (PDPPage page, AGMPortPtr portP,
    TextServer textServer, ASFixed scale, ASUns32 flags,
    ASFixedRect* updateRect, CancelProc cancelProc,
    void* cancelProcClientData);
```

| | |
|--------------------|---|
| Description | Draws a <i>PDPPage</i> to an <i>AGMPort</i> . Uses the current settings of the <i>AGMPort</i> , such as the current transformation matrix and the anti-alias setting. |
| Parameters | <p><i>page</i></p> <p>The page to render.</p> <p><i>portP</i></p> <p>The <i>AGMPort</i> into which the page is rendered.</p> <p><i>textServer</i></p> <p>Currently unused. Must be set to zero.</p> <p><i>scale</i></p> <p>Determines whether text is greeked or not. It is <i>not</i> used to perform scaling. To perform scaling, use AGMConcat on the <i>AGMPort</i> before calling this method.</p> <p><i>flags</i></p> <p>A bit field of PDPPageDrawFlags. Must be an OR of the following flags:</p> <p><i>kPDPPageDoLazyErase</i> — If set, erases the port if the first object drawn does not cover the page's entire crop box.</p> <p><i>kPDPPageUseAnnotFaces</i> — If set, draws annotations that have a default face, such as the visible fields in an Acrobat Form. Text and link annotations are <i>not</i> drawn.</p> <p><i>kPDPPageIsPrinting</i> — Should be set if the busy device drawn to is a printer. For example, under Windows, set this flag if the <i>HDC</i> for a printer is used in the call to AGMNewBitmapPort.</p> |

updateRect

A rectangle that specifies the region to redraw. Its coordinates are specified in unrotated page space. Any object outside *updateRect* is not passed to the *AGMPort* for rendering. May be *NULL*.

cancelProc

A method to call to check whether drawing should be canceled. If the method returns *true*, drawing is stopped; nothing is erased, and the port contains whatever was drawn up to the current state.

cancelProcClientData

User-supplied data to pass to *cancelProc* each time it is called.

Return Value None

Exceptions None

Notifications None

Header File *PXCalls.h*

Related Methods [AGMNewWindowPort](#)
[AGMSetAntiAliasPolicy](#)
[CCSetSystemCalibration](#)
[PDPageDrawContentsPlaced](#)
[PDPageDrawContentsToWindow](#)

Available in:

| Library API | 1.0 | 4.0 |
|-------------------|---------|---------|
| Adobe PDF Library | M, W, U | M, W, U |

| Plug-in API | 3.0, 3.01 | 4.0 |
|-------------|-----------|-----|
| Acrobat | — | — |
| Reader | — | — |

PDPageDrawContentsPlaced

```
void PDPageDrawContentsPlaced(PDPage page, AGMPortPtr portP,
    TextServer textServer, ASFixed scale, ASUns32 flags,
    ASFixedRect* updateRect, CancelProc cancelProc,
    void* cancelProcClientData);
```

Description

Draws a *PDPage* to an *AGMPort*. Uses the current settings of the *AGMPort*, such as the current transformation matrix and the anti-alias setting.

Note that *scale* is descriptive. It is used to determine whether text should be greeked. It is not used to perform scaling. To perform scaling, use [AGMConcat](#) on the *AGMPort* before calling this method.

This method is the same as [PDPageDrawContents](#), except that it raises an exception if the *pdPermCopy* permission is not set in the document.

Parameters

page

The page to render.

portP

The *AGMPort* into which the page is rendered.

textServer

Currently unused. Must be set to zero.

scale

Determines whether text is greeked or not.

flags

A bit field of [PDPageDrawFlags](#).

updateRect

A rectangle that specifies the region to redraw. Its coordinates are specified in unrotated page space. Any object outside *updateRect* is not passed to the *AGMPort* for rendering. May be *NULL*.

cancelProc

A method to call to check whether drawing should be canceled. If the method returns *true*, drawing is stopped; nothing is erased, and the port contains whatever was drawn up to the current state.

cancelProcClientData

User-supplied data to pass to *cancelProc* each time it is called.

| | |
|------------------------|--|
| Return Value | None |
| Exceptions | Raises pdErrOpNotPermitted if the <i>pdPermCopy</i> permission in PDPerms for the document is not set. |
| Notifications | None |
| Header File | <i>PXCalls.h</i> |
| Related Methods | PDPageDrawContents PDPageDrawContentsToWindow |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|---------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

PDPageDrawContentsPlacedToWindow

```
void PDPageDrawContentsPlacedToWindow(PDPage page,  
    void* window, void* displayContext, ASBool isDPS,  
    ASFixedMatrix* matrix, ASFixedRect* updateRect,  
    CancelProc cancelProc, void* cancelProcClientData);
```

Description

Draws the page to the window or display context. The window and display context are implementation dependent.

This method is the same as [PDPageDrawContentsToWindow](#), except that it raises an exception if the *pdPermCopy* permission is not set in the document.

Parameters

page

The page to draw into *window*.

window

The platform window to render to.

In Mac OS, the window is a *WindowPtr* or *GrafPtr* or *C GrafPtr*.

In Windows, the window is an *HWND*.

displayContext

The platform display context to render to.

In Mac OS, currently unused.

In Windows, *displayContext* is an *HDC*.

isDPS

Currently unused. Always set to *false*.

matrix

Pointer to the matrix to concatenate onto the default page matrix. It is useful for converting from page to window coordinates and for scaling.

updateRect

Pointer to the rectangle to draw, defined in user space coordinates. Any objects outside of *updateRect* will not be drawn. All objects are drawn if *updateRect* is *NULL*.

cancelProc

A method called to check whether drawing should be canceled. If the method returns *true*, drawing is stopped, nothing is erased, and the window contains whatever was drawn up to the current state.

cancelProcClientData

Pointer to user-supplied data to pass to *cancelProc* each time it is called. Should be *NULL* if *cancelProc* is *NULL*.

| | |
|------------------------|--|
| Return Value | None |
| Exceptions | Raises pdErrOpNotPermitted if the <i>pdPermCopy</i> permission in PDPerms for the document is not set. |
| Notifications | None |
| Header File | <i>PXCalls.h</i> |
| Related Methods | PDPageDrawContents PDPageDrawContentsToWindow |
| Availability | Plug-ins: Available if <i>PI_PDMODEL_VERSION</i> (in <i>PIRequir.h</i>) is set to 0x00020000 or higher. |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |
| Reader | — | — |

PDPAGEEMITPSORIENT

```
void PDPAGEEMITPSORIENT (PDPAGE pdPage, ASINT16 paperHeight,
    ASINT16 paperWidth, ASSTM stm, ASBOOL shrinkToFit,
    ASBOOL centerCropBox);
```

| | |
|------------------------|---|
| Description | Emits PostScript language code to crop and rotate a page. This method is only useful in the <i>PageBegin</i> callback in the <i>PDPrintClientRec</i> . |
| Parameters | <p><i>pdPage</i> The page to print.</p> <p><i>paperHeight</i> Height of the target paper (in points).</p> <p><i>paperWidth</i> Width of the target paper (in points).</p> <p><i>stm</i> The stream into which the PostScript language code can be written.</p> <p><i>shrinkToFit</i> If <i>true</i>, pages that are too large to print are shrunk until they fit on the paper.</p> <p><i>centerCropBox</i> If <i>true</i>, the crop box is centered on the paper.</p> |
| Return Value | None |
| Exceptions | None |
| Notifications | None |
| Header File | <i>PXCalls.h</i> |
| Related Methods | PDDocPrintPages |

Available in:

| | | |
|-------------------|-----------|---------|
| Library API | 1.0 | 4.0 |
| Adobe PDF Library | M, W, U | M, W, U |
| Plug-in API | 3.0, 3.01 | 4.0 |
| Acrobat | — | — |

| | | |
|-------------|-----------|-----|
| Plug-in API | 3.0, 3.01 | 4.0 |
| Reader | — | — |

Callbacks

ActivateProcType

```
ACCB1 void ACCB2 ActivateProcType (AVTool tool,  
    ASBool persistent);
```

| | |
|--------------------------|--|
| Description | Callback for AVTool . Called when this tool has become the active tool. It is legal to call AVAppGetActiveTool from within this method or from DeactivateProcType ; it will return this tool object. Call AVAppGetLastActiveTool to get the formerly active tool object (its DeactivateProcType method will already have been called). |
| Parameters | <p><i>tool</i></p> <p>The tool to activate.</p> <p><i>persistent</i></p> <p><i>true</i> if it should remain active through arbitrarily many “operations” (whatever that means for a particular tool), rather than performing a single action (“one shot”) and then restoring the previously active tool, <i>false</i> otherwise.</p> |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | DeactivateProcType |

AdjustCursorProcType

```
ACCB1 ASBool ACCB2 AdjustCursorProcType (AVTool tool,  
    AVPageView pageView, ASInt16 x, ASInt16 y);
```

| | |
|------------------------|--|
| Description | <i>(Optional)</i> Callback for AVTool . This callback controls the cursor shape when the tool is the active tool. If omitted, the cursor specified in the tool's cursorID field is used. |
| Parameters | <i>tool</i> The currently active tool. <i>pageView</i> The pageview in which the cursor is currently located. <i>x</i> The x-coordinate of the current cursor location. <i>y</i> The y-coordinate of the current cursor location. |
| Return Value | <i>true</i> if you handled the cursor shape (that is, do not allow underlying layers to handle it), <i>false</i> if you did (that is, allow underlying layers to handle it). |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVAppRegisterTool |

AGMMemAllocator

```
void* AGMMemAllocator (uint32 size, void* userData);
```

Description

Callback for [AGMMemObj](#). Allocates a specified amount of memory.

The memory block must be a *malloc*-style, non-relocatable block. It must respect any alignment rules for the processor on which the application is running. For example, when running on a PowerPC processor, which has 4-byte alignment limitations, you must return a block aligned on a 4-byte boundary.

In Mac OS:

1. If the memory is supplied by *NewHandle*, you must lock the memory by calling *HLock*. You cannot return **NewHandle* and leave the handle unlocked.
2. You must also call *StripAddress* (if it is available) on the address you return. That is, the returned address must be valid if *SwapMMUMode* is called.

Parameters

size

The number of bytes of memory to allocate.

userData

User-supplied data that was placed in the [AGMMemObj](#) structure.

Return Value

Pointer to the newly-allocated memory. This pointer must be non-zero if allocation succeeded. Returns 0 if allocation failed.

Header File

AGMemObj.h

Related Callbacks

[AGMMemDeleter](#)

Related Methods

[AGMNewBitmapPort](#)
[AGMNewRasterDev](#)
[AGMNewRasterPort](#)
[AGMNewWindowPort](#)

AGMMemDeleter

```
void AGMMemDeleter (void* ptr, void* userData);
```

| | |
|--------------------------|--|
| Description | Callback for AGMMemObj . Frees a specified block of memory that was allocated by AGMMemAllocator . This method is only called with pointers allocated by AGMMemAllocator and it is never called with 0 as the pointer. |
| Parameters | <p><i>ptr</i></p> <p>Pointer to the block of memory to free.</p> <p><i>userData</i></p> <p>User-supplied data that was placed in the AGMMemObj structure.</p> |
| Return Value | None |
| Header File | <i>AGMemObj.h</i> |
| Related Callbacks | AGMMemAllocator |
| Related Methods | AGMNewBitmapPort AGMNewRasterDev AGMNewRasterPort AGMNewWindowPort |

AGMPortDestructProcPtr

```
void AGMPortDestructProcPtr (void* userData);
```

| | |
|--------------------------|--|
| Description | Callback for AGMNewRasterPort . To be called when a AGM port for a raster device is destroyed. |
| Parameters | <i>userData</i> Always <i>NULL</i> . |
| Return Value | None |
| Header File | <i>AGMTypes.h</i> |
| Related Callbacks | None |
| Related Methods | AGMNewRasterPort |

ASExtensionEnumProc

```
ACCB1 ASBool ACCB2 ASExtensionEnumProc (ASExtension extension,  
void* clientData);
```

| | |
|--------------------------|--|
| Description | Enumeration function for ASEnumExtensions . |
| Parameters | <i>extension</i> The <i>ASExtension</i> for a plug-in. <i>clientData</i> User-supplied data that was passed in the call to ASEnumExtensions . |
| Return Value | If <i>false</i> , enumeration halts and the <i>ASExtension</i> on which enumeration halted is returned. If <i>true</i> , enumeration continues. |
| Header File | <i>CorExpt.h</i> |
| Related Callbacks | None |
| Related Methods | ASEnumExtensions |

ASFileCompletionProc

```
ACCB1 void ACCB2 ASFileCompletionProc (ASFile aFile,
    const char* p, ASInt32 fileOffsetRequested,
    ASInt32 countRequested, ASInt32 nBytesRead, ASInt32 error,
    void* compProcClientData);
```

| | |
|--------------------------|--|
| Description | Called when an asynchronous read or write request has completed. |
| Parameters | <p><i>aFile</i></p> <p>The <i>ASFile</i> for which data is read or written.</p> <p><i>p</i></p> <p>Pointer to the buffer provided by the client. Contains <i>nBytesRead</i> of data if <i>error</i> is zero.</p> <p><i>fileOffsetRequested</i></p> <p>File offset requested by the client.</p> <p><i>countRequested</i></p> <p>Number of bytes requested by the client.</p> <p><i>nBytesRead</i></p> <p>Number of bytes actually read or written.</p> <p><i>error</i></p> <p>Error condition if nonzero; see <i>AcroErr.h</i>.</p> <p><i>compProcClientData</i></p> <p>Client data parameter provided by client.</p> |
| Return Value | None |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | ASFileSysAsyncAbortProc ASFileSysAsyncReadProc ASFileSysAsyncWriteProc ASFileSysYieldProc |

ASFileSysAcquireFileSysPathProc

```
ACCB1 ASPathName ACCB2 ASFileSysAcquireFileSysPathProc  
(ASPathName pathName, ASFileSys newFileSys);
```

Description

Callback for [ASFileSysRec](#). Used for non-local file systems. Returns an *ASPathName* on the new *ASFileSys* that refers to an image (possibly cached) of the remote file. Because of the possibility of cache flushing, you must hold a copy of the remote file's *ASPathName* for the duration of use of the local file.

Note: Do not remove the local file copy, since the default file system does not know about the linkage to the remote file. Removing this temporary file is left to the file system.

Note: The [ASPathName](#) returned should be released with the [ASFileSysReleasePath](#) method when it is no longer needed.

Parameters

pathName

The *ASPathName* for which an equivalent in *newFileSys* is obtained.

newFileSys

The file system in which an equivalent of *pathName* is obtained.

Return Value

The *ASPathName* (in *newFileSys*) for the specified file. Returns *NULL* if one can not be made.

Header File

ASExpT.h

Related Callbacks

[ASFileSysCreatePathNameProc](#)

ASFileSysAsyncAbortProc

```
ACCB1 void ACCB2 ASFileSysAsyncAbortProc (MDFile file);
```

| | |
|--------------------------|--|
| Description | <p>Callback for ASFileSysRec. Aborts all uncompleted asynchronous I/O requests for the specified file. This callback can be called at any time.</p> <p>This callback calls each outstanding ASIORequest's ASIODoneProc to be called with the <i>totalBytes</i> = 0 and error = -1.</p> |
| Parameters | <p><i>file</i></p> <p>The file for which all uncompleted asynchronous read/write requests are aborted.</p> |
| Return Value | None |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | ASFileSysAsyncReadProc ASFileSysAsyncWriteProc ASFileSysYieldProc |

ASFileSysAsyncReadProc

```
ACCB1 ASInt32 ACCB2 ASFileSysAsyncReadProc (ASIORequest req);
```

Description

Callback for [ASFileSysRec](#). Asynchronously reads the specified data, returning immediately after the request has been queued. The *ASFileSys* must call the [ASIODoneProc](#) (if one was provided) when the specified data has been read.

This callback is similar to the [ASFileSysMReadRequestProc](#), except that this callback contains a caller-provided [ASIODoneProc](#), and can only be used for a single byte range.

Parameters

req

Data structure specifying the data to read. Contains information about the request including the *MDFile*, the file offset, the buffer for the request, the number of bytes in the request and an [ASIODoneProc](#) and *clientData* for the [ASIODoneProc](#).

If the [ASIODoneProc](#) in *req* is non-*NULL*, and there is an error queueing the read request, the [ASIODoneProc](#) must *not* be called.

Return Value

0 if the request was successfully queued, otherwise returns a nonzero platform-dependent error code.

Header File

ASExpT.h

Related Callbacks

[ASFileSysAsyncAbortProc](#)
[ASFileSysAsyncWriteProc](#)
[ASFileSysYieldProc](#)

ASFileSysAsyncWriteProc

```
ACCB1 ASInt32 ACCB2 ASFileSysAsyncWriteProc (ASIORequest req);
```

| | |
|--------------------------|---|
| Description | Callback for ASFileSysRec . Asynchronously writes the specified data, returning immediately after the request has been queued. The <i>ASFileSys</i> must call the ASIODoneProc (if one was provided) when the specified data has been written. |
| Parameters | <i>req</i> Data structure specifying the data to write. Contains information about the request including the <i>MDFile</i> , the file offset, the buffer for the request, the number of bytes in the request and an ASIODoneProc and <i>clientData</i> for the ASIODoneProc . If the ASIODoneProc in <i>req</i> is non- <i>NULL</i> , and there is an error queueing the write request, the ASIODoneProc must <i>not</i> be called. |
| Return Value | 0 if the request was successfully queued, otherwise returns a nonzero platform-dependent error code. |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | ASFileSysAsyncAbortProc ASFileSysAsyncReadProc ASFileSysYieldProc |

ASFileSysClearOutstandingMReadsProc

```
ACCB1 void ACCB2 ASFileSysClearOutstandingMReadsProc  
    (MDFile file);
```

| | |
|--------------------------|--|
| Description | Callback for ASFileSysRec . Used to advise a file system that the previous range of bytes requested to read are not needed, so that it may drop the read requests. The file system can continue pushing the bytes if it cannot stop the reads. |
| Parameters | <i>file</i> The file that was being read. |
| Return Value | None |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | ASFileSysMReadRequestProc |
| Related Methods | ASFileRead |

ASFileSysCloseProc

```
ACCB1 ASInt32 ACCB2 ASFileSysCloseProc (MDFile f);
```

| | |
|--------------------------|--|
| Description | Callback for ASFileSysRec . This callback is responsible for closing the specified file. It is called by ASFileClose . |
| Parameters | <i>f</i> The file to close. |
| Return Value | 0 if the request was successful, otherwise returns a nonzero platform-dependent error code. |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | ASFileSysOpenProc |
| Related Methods | ASFileClose |

ASFileSysCopyPathNameProc

```
ACCB1 ASPathName ACCB2 ASFileSysCopyPathNameProc  
(ASPathName pathName) ;
```

Description

Callback for [ASFileSysRec](#). Copies a pathname (not the underlying file). It is called by [ASFileSysCopyPath](#).

Copying a pathname does not result in any file-level operations, and is unaffected by whether there is or is not an open file for the path name.

Note: The [ASPathName](#) returned should be released by the [ASFileSysReleasePath](#) method when it is no longer needed.

Parameters

pathName

The pathname to copy.

Return Value

New copy of the pathname.

Header File

ASExpT.h

Related Methods

[ASFileSysCopyPath](#)

ASFileSysCreatePathNameProc

```
ACCB1 ASPathName ACCB2 ASFileSysCreatePathNameProc  
(ASAtom pathSpecType, const void* pathSpec,  
const void* mustBeZero);
```

Description

Callback for [ASFileSysRec](#). Creates an [ASPathName](#) based on the input type and [PDFileSpec](#). Each [ASFileSys](#) implementation must publish the input types that it accepts. For example, the Macintosh [ASFileSys](#) may accept types of "SFReply *" or "FSSpecPtr," and the MS-DOS [ASFileSys](#) may only accept types of "CString."

Note: The [ASPathName](#) returned should be released by the [ASFileSysReleasePath](#) method when it is no longer needed.

Parameters

specType

The type of the input [PDFileSpec](#).

fileSpec

The file specification from which to create an [ASPathName](#).

mustBeZero

Reserved for future use.

Return Value

The newly-created pathname.

Header File

[ASExpT.h](#)

Related Methods

[ASFileSysCreatePathName](#)

ASFileSysDiPathFromPathProc

```
ACCB1 char* ACCB2 ASFileSysDiPathFromPathProc (ASPathName path,  
ASPathName relativeToThisPath);
```

Description

Callback for [ASFileSysRec](#). Converts a pathname to a device-independent pathname. It is called by [ASFileSysDiPathFromPath](#).

Note: The memory for the char returned should be freed with the [ASfree](#) method when it is no longer needed.*

Parameters

path

The pathname to convert to a device-independent pathname.

relativeToThisPath

The path relative to which the device-independent pathname is specified. Pass *NULL* if the device-independent pathname (for example, c:\dir1\dir2\dir3\myfile.pdf) is an absolute pathname instead of a relative pathname (for example, ../../dir3/myfile.pdf).

Return Value

The device-independent pathname.

Header File

ASExpT.h

Related Methods

[ASFileSysDiPathFromPath](#)

ASFileSysDisposePathNameProc

```
ACCB1 void ACCB2 ASFileSysDisposePathNameProc  
    (ASPathName pathName) ;
```

| | |
|------------------------|---|
| Description | Callback for ASFileSysRec . It is called by ASFileSysReleasePath . This callback frees any memory occupied by a pathname. It does not result in any file-level operations. |
| Parameters | <i>pathName</i> The pathname to release. |
| Return Value | None |
| Header File | <i>ASExpT.h</i> |
| Related Methods | ASFileSysReleasePath |

ASFileSysFlushProc

```
ACCB1 ASInt32 ACCB2 ASFileSysFlushProc (MDFile f);
```

| | |
|--------------------------|--|
| Description | Callback for ASFileSysRec . Flushes data for the specified file. It is called by ASFileFlush . |
| Parameters | <i>f</i> The file to flush. |
| Return Value | 0 if the request was successful, otherwise returns a nonzero platform-dependent error code. |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | ASFileSysFlushVolumeProc |
| Related Methods | ASFileFlush |

ASFileSysFlushVolumeProc

```
ACCB1 ASInt32 ACCB2 ASFileSysFlushVolumeProc  
(ASPathName pathName);
```

| | |
|--------------------------|--|
| Description | <p>Callback for ASFileSysRec. Flushes the volume on which the specified file resides. This ensures that any data written to the system for the volume containing <i>pathName</i> is flushed out to the physical volume (equivalent to the Macintosh <i>FlushVol</i>, or to the UNIX <i>sync</i>). Call this after you're finished writing a complete "transaction" to force a commit.</p> <p>This callback is not called directly from any plug-in API method, but is used internally by the Acrobat viewer.</p> |
| Parameters | <p><i>pathName</i></p> <p>The pathname for the file whose volume is flushed.</p> |
| Return Value | 0 if the request was successful, otherwise returns a nonzero platform-dependent error code. |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | ASFileSysFlushProc |

ASFileSysGetEofProc

```
ACCB1 ASInt32 ACCB2 ASFileSysGetEofProc (MDFFile f,  
ASUns32* pos);
```

| | |
|------------------------|---|
| Description | Callback for ASFileSysRec . Gets a file's current logical size. Called by ASFileGetEOF . |
| Parameters | <i>f</i> The file whose logical size is obtained. <i>pos</i> (Filled by the callback) The file's logical size, in bytes. |
| Return Value | 0 if the request was successful, otherwise returns a nonzero platform-dependent error code. |
| Header File | <i>ASExpT.h</i> |
| Related Methods | ASFileGetEOF ASFileSetEOF |

ASFileSysGetFileFlags

```
ACCB1 ASUns32 ACCB2 ASFileSysGetFileFlags (MDFile file);
```

| | |
|--------------------------|--|
| Description | Callback for ASFileSysRec . Gets the flags for the specified file. |
| Parameters | <i>file</i> The file whose flags are obtained. |
| Return Value | Bit field using ASFile Flags . |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | None |

ASFileSysGetFileSysNameProc

ACCB1 [ASAtom](#) ACCB2 ASFileSysGetFileSysNameProc (void);

| | |
|------------------------|---|
| Description | Callback for ASFileSysRec . Gets this file system's name. This callback is not called directly by any method in the plug-in API, but is used internally by the Acrobat viewer. |
| Parameters | None |
| Return Value | The <i>ASAtom</i> containing the name of this file system. |
| Header File | <i>ASExpT.h</i> |
| Related Methods | ASFileRegisterFileSys |

ASFileSysGetNameProc

```
ACCB1 ASInt32 ACCB2 ASFileSysGetNameProc (ASPathName pathName,  
char* name, ASInt32 maxLength);
```

| | |
|--------------------------|---|
| Description | <p>Callback for ASFileSysRec. Returns a character string containing the file name for the specified <i>ASPathName</i>. The character string contains only the file name; it is not a complete pathname.</p> <p>This callback is not called directly from any plug-in API method. It is used internally by the Acrobat viewer.</p> |
| Parameters | <p><i>pathName</i></p> <p>The <i>ASPathName</i> for which the file name is returned.</p> <p><i>name</i></p> <p>(Filled by the callback) Character string containing the file name for <i>pathName</i>.</p> <p><i>maxLength</i></p> <p>Maximum number of characters that <i>name</i> can hold.</p> |
| Return Value | 0 if the request was successful, otherwise returns a nonzero platform-dependent error code. |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | ASFileSysGetFileSysNameProc |

ASFileSysGetPosProc

```
ACCB1 ASInt32 ACCB2 ASFileSysGetPosProc (MDFile f,  
ASUns32* pos);
```

| | |
|--------------------------|--|
| Description | Gets the current position for the specified file. Called by ASFileGetPos . |
| Parameters | <i>f</i> The file whose current position is obtained. <i>pos</i> (Must be filled by the callback) The current position. |
| Return Value | 0 if the request was successful, otherwise returns a nonzero platform-dependent error code. |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | ASFileSysSetPosProc |
| Related Methods | ASFileGetPos ASFileSetPos |

ASFileSysGetStatusProc

```
ACCB1 ASUns32 ACCB2 ASFileSysGetStatusProc (MDFile file);
```

| | |
|------------------------|---|
| Description | Callback for ASFileSysRec . Gets the status of the specified file. This callback is used for asynchronous I/O. For example, it can indicate that an underlying file connection has been closed. |
| Parameters | <i>file</i> The file whose status is obtained. |
| Return Value | The file's status. Must be one of the ASFileStatus Flags values. |
| Header File | <i>ASExpT.h</i> |
| Related Methods | ASFileRead |

ASFileSysGetStorageFreeSpaceProc

```
ACCB1 ASUns32 ACCB2 ASFileSysGetStorageFreeSpaceProc  
( ASPathName pathName );
```

| | |
|--------------------------|--|
| Description | Callback for ASFileSysRec . Gets the amount of free space on the volume containing the specified <i>ASPathName</i> . |
| Parameters | <i>pathName</i> The <i>ASPathName</i> for a file on the volume whose free space is obtained. |
| Return Value | Free space, in bytes. Because the free space is returned as an <i>ASUns32</i> , it is limited to 4 GB. |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | None |

ASFileSysGetTempPathNameProc

```
ACCB1 ASPathName ACCB2 ASFileSysGetTempPathNameProc  
(ASPathName pathName) ;
```

| | |
|--------------------------|---|
| Description | Callback for ASFileSysRec . Returns an unique pathname suitable for use in creating temporary files. <i>Note: The ASPathName returned should be released by the ASFileSysReleasePath method when it is no longer needed.</i> |
| Parameters | <i>pathName</i> If <i>pathName</i> is non- <i>NULL</i> , the temporary file must be stored such that a rename of <i>pathName</i> will succeed (for example, on the same volume if renames across volumes are illegal on this file system). |
| Return Value | Pathname for a temporary file. |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | ASFileSysCopyPathNameProc |

ASFileSysIsSameFileProc

```
ACCB1 ASBool ACCB2 ASFileSysIsSameFileProc (MDFile f,  
      ASPathName pathName, ASPathName newPathName);
```

| | |
|--------------------------|--|
| Description | Callback for ASFileSysRec . Tests whether two files are the same. |
| Parameters | <p><i>f</i></p> <p>The <i>ASFile</i> of first file to compare (in many implementations, it may be unnecessary to use this information, <i>pathName</i> and <i>newPathName</i> provide sufficient information).</p> <p><i>pathName</i></p> <p>The <i>ASPathName</i> of first file to compare.</p> <p><i>newPathName</i></p> <p>The <i>ASPathName</i> of second file to compare.</p> |
| Return Value | 0 if the files are different, nonzero if they are the same. |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | None |

ASFileSysMReadRequestProc

```
ACCB1 ASInt32 ACCB2 ASFileSysMReadRequestProc (MDFile file,  
ASFile aFile, ASInt32* blockPairs, ASInt32 nBlockPairs);
```

Description

Callback for [ASFileSysRec](#). Queues asynchronous requests for one or more byte ranges that the caller (usually the Acrobat viewer or Library) will need in the near future. This callback is important for slow file systems, such as the Web, to improve overall performance by allowing the file system to begin retrieving bytes before they are actually needed, while the Acrobat software continues processing as much as it can with the data that has already been downloaded.

This callback does not actually read the data, but merely queues the requests, starts the asynchronous code that reads the data, and returns. The asynchronous code that reads the data must use [ASFilePushData](#) to push the data from each byte range to the Acrobat software as soon as the data is ready.

This callback is similar to the [ASFileSysAsyncReadProc](#), except that this callback contains a caller-provided [ASIODoneProc](#), and can only be used for a single byte range.

Parameters

file

The file whose data is read.

aFile

The corresponding *ASFile*, for use with [ASFilePushData](#).

blockPairs

An array of file offsets and byte lengths.

nBlockPairs

The number of block pairs in *blockPairs*.

Return Value

0 if the request was successfully queued, otherwise returns a nonzero platform-dependent error code.

Header File

ASExpT.h

Related Callbacks

[ASFileSysAsyncReadProc](#)

Related Methods

[ASFileRead](#)

ASFileSysOpenProc

```
ACCB1 ASInt32 ACCB2 ASFileSysOpenProc (ASPathName pathName,  
    ASUns16 mode, MDFile* fP);
```

| | |
|--------------------------|--|
| Description | Callback for ASFileSysRec . Opens the specified file. It is called by ASFileSysOpenFile and ASFileReopen . |
| Parameters | <p><i>pathName</i></p> <p>The pathname for the file to open.</p> <p><i>mode</i></p> <p>The mode in which the file is opened. Must be an OR of the ASFile Open Modes.</p> <p><i>fP</i></p> <p>(Filled by the callback) The newly-opened file.</p> |
| Return Value | 0 if the request was successful, otherwise returns a nonzero platform-dependent error code. |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | ASFileSysCloseProc |
| Related Methods | ASFileSysOpenFile ASFileReopen ASFileClose |

ASFileSysPathFromDIPathProc

```
ACCB1 ASPathName ACCB2 ASFileSysPathFromDIPathProc  
(char* diPath, ASPathName relativeToThisPath);
```

Description

Callback for [ASFileSysRec](#). Converts a device-independent pathname to an *ASPathName*. It is called by [ASFileSysPathFromDIPath](#).

Note: The [ASPathName](#) returned should be released by the [ASFileSysReleasePath](#) method when it is no longer needed.

Parameters

diPath

Device-independent pathname to convert to an *ASPathName*.

relativeToThisPath

If *diPath* is an absolute pathname, *NULL*.

If *diPath* is a relative pathname, the pathname relative to which it is specified.

Return Value

The *ASPathName* corresponding to the specified device-independent pathname.

Header File

ASExpT.h

Related Methods

[ASFileSysPathFromDIPath](#)
[ASFileSysDIPathFromPath](#)

ASFileSysReadProc

```
ACCB1 ASSize_t ACCB2 ASFileSysReadProc (void* ptr,  
    ASSize_t size, ASSize_t count, MDFile f, ASInt32* pError);
```

| | |
|--------------------------|--|
| Description | Callback for ASFileSysRec . Reads data from the specified file. It is called by ASFileRead . |
| Parameters | <p><i>ptr</i></p> <p><i>(Filled by the callback)</i> The data read from the file.</p> <p><i>size</i></p> <p>The size of each item to read.</p> <p><i>count</i></p> <p>The number of items to read.</p> <p><i>f</i></p> <p>The file from which data is read.</p> <p><i>pError</i></p> <p><i>(Filled by the callback)</i> Error code. This value is filled only if an error occurred. In that case, it should contain an error code.</p> |
| Return Value | The number of bytes read, or 0 if there was an error. |
| Header File | ASExpT.h |
| Related Callbacks | ASFileSysWriteProc |
| Related Methods | ASFileRead |

ASFileSysRemoveProc

```
ACCB1 ASInt32 ACCB2 ASFileSysRemoveProc (ASPathName pathName);
```

| | |
|------------------------|---|
| Description | Callback for ASFileSysRec . Deletes a file. It is called by ASFileSysRemoveFile . |
| Parameters | <i>pathName</i> The file to delete. |
| Return Value | 0 if the request was successful, otherwise returns a nonzero platform-dependent error code. |
| Header File | <i>ASExpT.h</i> |
| Related Methods | ASFileSysRemoveFile |

ASFileSysRenameProc

```
ACCB1 ASInt32 ACCB2 ASFileSysRenameProc (MDFile* f,  
    ASPathName oldPath, ASPathName newPath);
```

| | |
|--------------------------|---|
| Description | Callback for ASFileSysRec . Renames a file. It is not called directly by any method in the plug-in API, but is used internally by the Acrobat viewer. |
| Parameters | <i>f</i> The file to rename. <i>oldPath</i> The file's old pathname. <i>newPath</i> The file's new pathname. |
| Return Value | 0 if the request was successful, otherwise returns a nonzero platform-dependent error code. |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | ASFileSysGetNameProc |

ASFileSysSetEofProc

```
ACCB1 ASInt32 ACCB2 ASFileSysSetEofProc (MDFile f, ASUns32 pos);
```

| | |
|------------------------|---|
| Description | Callback for ASFileSysRec . Increases or decreases the logical size of a file. It is called by ASFileSetEOF . |
| Parameters | <i>f</i> The file to expand/shrink. <i>pos</i> The desired size, in bytes. |
| Return Value | 0 if the request was successful, otherwise returns a nonzero platform-dependent error code. |
| Header File | <i>ASExpT.h</i> |
| Related Methods | ASFileGetEOF ASFileSetEOF |

ASFileSysSetPosProc

```
ACCB1 ASInt32 ACCB2 ASFileSysSetPosProc (MDFFile f, ASUns32 pos);
```

| | |
|------------------------|--|
| Description | Callback for ASFileSysRec . Sets the current position in a file (that is, the point from which data will next be read). It is called by ASFileSetPos . |
| Parameters | <i>f</i> The file in which the position is set. <i>pos</i> The desired new position (specified in bytes from the beginning of the file). |
| Return Value | 0 if the request was successful, otherwise returns a nonzero platform-dependent error code. |
| Header File | <i>ASExpT.h</i> |
| Related Methods | ASFileGetPos ASFileSetPos |

ASFileSysWriteProc

```
ACCB1 ASSize_t ACCB2 ASFileSysWriteProc (void* ptr,  
    ASSize_t size, ASSize_t count, MDFile f, ASInt32* pError);
```

| | |
|--------------------------|--|
| Description | Callback for ASFileSysRec . Writes data to the specified file. |
| Parameters | <p><i>ptr</i></p> <p>Buffer containing data to write.</p> <p><i>size</i></p> <p>The size of each item to write.</p> <p><i>count</i></p> <p>The number of items to write.</p> <p><i>f</i></p> <p>The file into which data is written.</p> <p><i>pError</i></p> <p>(Filled</p> |
| Return Value | The number of bytes written. Returns 0 if there was an error. |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | ASFileSysReadProc |
| Related Methods | ASFileWrite |

ASFileSysYieldProc

```
ACCB1 ASInt32 ACCB2 ASFileSysYieldProc (MDFile file);
```

| | |
|--------------------------|---|
| Description | <p>Callback for ASFileSysRec. Yields on the asynchronous I/O requests for the specified file to allow other processes a chance to process events that may be required for a file read to complete. An <i>ASFileSys</i> should implement a yield mechanism to complement asynchronous read/write requests.</p> <p>In Windows, this could be a normal <i>PeekMessage</i> based yield.</p> <p>In UNIX, it could mean using <i>select</i> on a file descriptor.</p> |
| Parameters | <p><i>file</i></p> <p>The file whose asynchronous I/O requests are yielded.</p> |
| Return Value | 0 if the request was successful, otherwise returns a nonzero platform-dependent error code. |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | ASFileSysAsyncAbortProc ASFileSysAsyncReadProc ASFileSysAsyncWriteProc |
| Related Methods | ASFileRead |

ASIODoneProc

```
ACCB1 void ACCB2 ASIODoneProc (ASIORequest req);
```

Description

Callback in [ASIORequest](#). Used by the asynchronous read/write *ASFileSys* implementation. Provided by the *ASFile* implementation to the *ASFileSys*. The *ASFileSys* must call this method when an asynchronous request is completed:

- When an I/O request has some or all of its data.
- If the request is successfully queued but an error prevents it from completing.
- If the request is aborted by calling [ASFileSysAsyncAbortProc](#). In this case, the *totalBytesCompleted*=0 and *pError*=-1.

If the request fails, this method must still be called, with the error. It is not called if there is an error queueing the read or write request, however.

Parameters

req

The I/O request for which data has been read/written.

Return Value

None

Header File

ASExpT.h

Related Callbacks

[ASFileSysAsyncAbortProc](#)
[ASFileSysAsyncReadProc](#)
[ASFileSysAsyncWriteProc](#)
[ASFileSysYieldProc](#)

ASMemAllocProc

```
void* ASMemAllocProc (void* clientData, ASSize_t size);
```

Description Called when the PDF Library needs to allocate memory. Callback should allocate the requested amount of memory and return it as a `void*`.

Parameters *clientData*
The *clientData* specified in the [TKAllocatorProcs](#) structure.

size
Amount of memory requested.

Return Value Pointer to allocated block of memory.

Header File *PDFInit.h*

Related Callbacks [ASMemAvailProc](#)
[ASMemFreeProc](#)
[ASMemReallocProc](#)

Related Methods [PDFLInit](#)

Example

```
void* StdMemAllocProc(void* clientData,
ASSize_t size)
{
return malloc(size);
}
```


ASMemAvailProc

```
ASSize_t* ASMemAvailProc (void* clientData);
```

Description Called when the PDF Library needs to determine the amount of available free memory.

Parameters *clientData*
The *clientData* specified in the [TKAllocatorProcs](#) structure.

Return Value Amount of available free memory.

Header File *PDFInit.h*

Related Callbacks [ASMemAllocProc](#)
[ASMemFreeProc](#)
[ASMemReallocProc](#)

Related Methods [PDFLInit](#)

Example

```
os_size_t sys_memavail(void)
{
    MEMORYSTATUS ms;

    ms.dwLength = sizeof(ms);
    GlobalMemoryStatus(&ms);

    return ms.dwAvailPhys + ms.dwAvailPageFile;
}
```

ASMemFreeProc

```
void* ASMemFreeProc (void* clientData, void* p);
```

Description Called when the PDF Library needs to free memory. Callback should free the requested block of memory.

Parameters *clientData*
The *clientData* specified in the [TKAllocatorProcs](#) structure.

p
Pointer to memory to free.

Return Value None

Header File *PDFInit.h*

Related Callbacks [ASMemAllocProc](#)
[ASMemAvailProc](#)
[ASMemReallocProc](#)

Related Methods [PDFLInit](#)

Example

```
void StdMemFreeProc(void* clientData,  
void* p)  
{  
    free(p);  
}
```

ASMemReallocProc

```
void* ASMemReallocProc (void* clientData, void* p,  
    ASSize_t size);
```

| | |
|--------------------------|--|
| Description | Called when the PDF Library needs to reallocate memory. Callback should reallocate the requested amount of memory and return it as a void*. |
| Parameters | <p><i>clientData</i></p> <p>The <i>clientData</i> specified in the TKAllocatorProcs structure.</p> <p><i>p</i></p> <p>Pointer to memory to reallocate.</p> <p><i>size</i></p> <p>Amount of memory requested.</p> |
| Return Value | Pointer to allocated block of memory. |
| Header File | <i>PDFInit.h</i> |
| Related Callbacks | ASMemAllocProc ASMemAvailProc ASMemFreeProc |
| Related Methods | PDFLInit |
| Example | <pre>void* StdMemReallocProc(void* clientData, void* p, ASSize_t size) { return realloc(p, size); }</pre> |

ASProcStmDestroyProc

```
ACCB1 void ACCB2 ASProcStmDestroyProc (void* clientData);
```

| | |
|--------------------------|--|
| Description | Callback for use by ASProcStmWrOpen . Called at end of stream so you can do clean up and free allocated memory. |
| Parameters | <i>clientData</i> User-supplied data that was passed in the call to ASProcStmWrOpen . |
| Return Value | None |
| Header File | <i>ASExpt.h</i> |
| Related Callbacks | ASStmProc |
| Related Methods | ASProcStmWrOpen |

ASStmProc

```
ACCB1 ASInt32 ACCB2 ASStmProc (char* data, ASInt32 nData,  
    void* clientData);
```

Description

Callback for use by [ASProcStmRdOpen](#) and [ASProcStmWrOpen](#). This procedure must return the number of bytes specified by *nData*, obtaining them in any way it wishes.

If your procedure reads data from a file, it is generally quite inefficient to open the file, read the bytes, then close the file each time bytes are requested. Instead, consider opening the file the first time bytes are requested from it, reading the entire file into a secondary buffer, and closing the file. When subsequent requests for data from the file are received, simply copy data from the secondary buffer, rather than re-opening the file.

Parameters

data

(Filled by the callback) Buffer into which your procedure must place the number of bytes specified by *nData*.

nData

Number of bytes to read from the stream and place into *data*.

clientData

User-supplied data that was passed in the call to [ASProcStmRdOpen](#) or [ASProcStmWrOpen](#).

Return Value

The number of bytes actually read or written.

Header File

ASExpt.h

Related Callbacks

[ASProcStmDestroyProc](#)

Related Methods

[ASProcStmRdOpen](#)
[ASProcStmWrOpen](#)

AVActionCopyProc

```
ACCB1 PDAction ACCB2 AVActionCopyProc (void* actionHandlerObj,  
    AVDoc fromDoc, PDAction anAction, AVDoc toDoc);
```

| | |
|------------------------|--|
| Description | <i>(Optional)</i> Callback for AVActionHandlerProcs . Called upon to copy an action, possibly to another document. Action handlers should provide this callback to allow for copying their actions. Called by AVDocCopyAnnot . |
| Parameters | <i>actionHandlerObj</i> User-supplied data that was passed when the action handler was registered using AVAppRegisterActionHandler . <i>fromDoc</i> The document whose action is copied. <i>anAction</i> The action to copy. <i>toDoc</i> The document to which the action is copied. |
| Return Value | The newly-created PDAction copy. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocCopyAction |

AVActionDoPropertiesProc

```
ACCB1 void ACCB2 AVActionDoPropertiesProc  
(void* actionHandlerObj, PDAction action, AVDoc doc);
```

| | |
|------------------------|--|
| Description | Callback for AVActionHandlerProcs . Displays a user interface that allows a user to set the action's properties (for example, for a <i>Launch</i> action, set the file to launch; for a <i>GoTo</i> action, select the destination page/zoom/coordinates). |
| Parameters | <p><i>actionHandlerObj</i></p> <p>User-supplied data that was passed when the action handler was registered using AVAppRegisterActionHandler.</p> <p><i>action</i></p> <p>The action whose properties are set.</p> <p><i>doc</i></p> <p>The document in which the action is located.</p> |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVActionHandlerGetProcs |

AVActionEnumProc

```
ACCB1 ASBool ACCB2 AVActionEnumProc (ASAtom type,  
    char* userName, void* clientData);
```

Description Callback used by [AVAppEnumActionHandlers](#). It is called once for each action handler.

Parameters *type*

The action handler's name.

userName

The action's name, as it appears in the Acrobat viewer's user interface.

clientData

User-supplied data that was passed in the call to [AVAppEnumActionHandlers](#).

Return Value Return *true* to continue enumeration, *false* to halt enumeration.

Header File *AVExpT.h*

Related Methods [AVAppEnumActionHandlers](#)

AVActionFillActionDictProc

```
ACCB1 void ACCB2 AVActionFillActionDictProc  
(void* actionHandlerObj, CosObj actionDict, AVDoc doc);
```

| | |
|--------------------------|---|
| Description | <i>(Required)</i> Callback for AVActionHandlerProcs . This required function is called as soon as the user selects the action from the action types pop-up menu. It allows an action handler to populate a newly-created action's <i>actionDict</i> . At the time this method is called, the information needed to completely specify an action is often not yet available. As a result, this method is generally a good place to populate the <i>actionDict</i> with default values. |
| Parameters | <i>actionHandlerObj</i> User-supplied data that was passed when the action handler was registered using AVAppRegisterActionHandler . <i>actionDict</i> Action dictionary to populate with default values. <i>doc</i> The document in which the action is located. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | None |

AVActionGetButtonTextProc

```
ACCB1 ASInt32 ACCB2 AVActionGetButtonTextProc  
(void* actionHandlerObj, PDAction action, char* buffer,  
ASInt32 bufLen, AVDoc doc);
```

| | |
|--------------------------|--|
| Description | <p>Callback for AVActionHandlerProcs. This optional function should store into buffer a <i>NULL</i>-terminated C string which is a localized string for the “edit action” button in the action dialog.</p> <p>For example, the Acrobat viewer’s built-in “OpenFile” action returns “Select File” for this string.</p> |
| Parameters | <p><i>actionHandlerObj</i></p> <p>User-supplied data that was passed when the action handler was registered using AVAppRegisterActionHandler.</p> <p><i>action</i></p> <p>The action whose button text is returned.</p> <p><i>buffer</i></p> <p>(Filled by the callback) The button text.</p> <p><i>bufLen</i></p> <p>Length of <i>buffer</i>, in bytes.</p> <p><i>doc</i></p> <p>The document in which the action is located.</p> |
| Return Value | The number of characters copied into <i>buffer</i> . |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVActionGetStringOneTextProc AVActionGetStringTwoTextProc |

AVActionGetInstructionsProc

```
ACCB1 ASInt32 ACCB2 AVActionGetInstructionsProc  
    (void* actionHandlerObj, PDAction action, char* buffer,  
    ASInt32 bufLen, AVDoc doc);
```

| | |
|--------------------------|---|
| Description | Callback for AVActionHandlerProcs . This optional function should store into buffer a <i>NULL</i> -terminated C string which contains localized instructions for the action creation/properties dialog. |
| Parameters | <p><i>actionHandlerObj</i></p> <p>User-supplied data that was passed when the action handler was registered using AVAppRegisterActionHandler.</p> <p><i>action</i></p> <p>The action whose instructions are returned.</p> <p><i>buffer</i></p> <p>(Filled by the callback) The instruction text.</p> <p><i>bufLen</i></p> <p>Length of <i>buffer</i>, in bytes.</p> <p><i>doc</i></p> <p>The document in which the action is located.</p> |
| Return Value | The number of characters copied into <i>buffer</i> . |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVActionPerformProc |

AVActionGetStringOneTextProc

```
ACCB1 ASInt32 ACCB2 AVActionGetStringOneTextProc  
    (void* actionHandlerObj, PDAction action, char* buffer,  
    ASInt32 bufLen, AVDoc doc);
```

| | |
|--------------------------|--|
| Description | <p>(<i>Optional</i>) Callback for AVActionHandlerProcs. This function should store into <i>buffer</i> a <i>NULL</i>-terminated C string which is a localized string placed above the “edit action” button in the action dialog. A <i>NULL</i> proc will cause the button to hide.</p> <p>For example, the Acrobat viewer’s built-in “OpenFile” action returns “File Name: <the current file name>” for this string.</p> |
| Parameters | <p><i>actionHandlerObj</i></p> <p>User-supplied data that was passed when the action handler was registered using AVAppRegisterActionHandler.</p> <p><i>action</i></p> <p>The action whose “string 1” text is returned.</p> <p><i>buffer</i></p> <p>(<i>Filled by the callback</i>) The string text appearing above the button.</p> <p><i>bufLen</i></p> <p>Length of <i>buffer</i>, in bytes.</p> <p><i>doc</i></p> <p>The document in which the action is located.</p> |
| Return Value | The number of characters copied into <i>buffer</i> . |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVActionGetButtonTextProc AVActionGetStringTwoTextProc |

AVActionGetStringTwoTextProc

```
ACCB1 ASInt32 ACCB2 AVActionGetStringTwoTextProc  
(void* actionHandlerObj, PDAction action, char* buffer,  
ASInt32 bufLen, AVDoc doc);
```

Description

Callback for [AVActionHandlerProcs](#). This optional function should store into *buffer* a *NULL*-terminated C string which is a localized string placed below the “edit action” button in the action dialog.

For example, the Acrobat viewer’s built-in “OpenFile” action returns nothing for this string, but the built-in “GoToView” action returns a description of the current zoom type.

Parameters

actionHandlerObj

User-supplied data that was passed when the action handler was registered using [AVAppRegisterActionHandler](#).

action

The action whose “string 2” text is returned.

buffer

(Filled by the callback) The string text appearing below the button.

bufLen

Length of *buffer*, in bytes.

doc

The document in which the action is located.

Return Value

The number of characters copied into *buffer*.

Header File

AVExpT.h

Related Callbacks

[AVActionGetButtonTextProc](#)
[AVActionGetStringOneTextProc](#)

AVActionPerformProc

```
ACCB1 void ACCB2 AVActionPerformProc (void* actionHandlerObj,  
    PDAction action, AVDoc doc);
```

| | |
|--------------------------|--|
| Description | Callback for AVActionHandlerProcs . Performs the action. |
| Parameters | <p><i>actionHandlerObj</i></p> <p>User-supplied data that was passed when the action handler was registered using AVAppRegisterActionHandler.</p> <p><i>action</i></p> <p>The action to perform.</p> <p><i>doc</i></p> <p>The document in which the action is located.</p> |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVActionGetInstructionsProc |

AVAnnotHandlerAdjustCursorProc

```
ACCB1 ASBool ACCB2 AVAnnotHandlerAdjustCursorProc  
(AVAnnotHandler annotHandler, PDAnnot anAnnot,  
AVPageView pageView, ASInt16 xHit, ASInt16 yHit);
```

| | |
|--------------------------|--|
| Description | (Optional) Callback for AVAnnotHandler . It controls the cursor shape when the cursor is within the annotation. If <i>NULL</i> , the annotation behaves as if the <i>AdjustCursor</i> callback returned <i>false</i> . |
| Parameters | <p><i>annotHandler</i></p> <p>The annotation handler responsible for this annotation.</p> <p><i>anAnnot</i></p> <p>The annotation containing the cursor.</p> <p><i>pageView</i></p> <p>The <i>AVPageView</i> in which the annotation is located.</p> <p><i>xHit</i></p> <p>The cursor's current x-coordinate.</p> <p><i>yHit</i></p> <p>The cursor's current y-coordinate.</p> |
| Return Value | <i>true</i> if the callback handled the adjust cursor event, <i>false</i> otherwise. The callback would return <i>false</i> , for example, if the annotation is irregularly shaped and the cursor is not currently over the real annotation even though it is within the rectangular bounding box that the Acrobat viewer uses to specify annotations. |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | None |

AVAnnotHandlerCopyProc

```
ACCB1 PDAnnot ACCB2 AVAnnotHandlerCopyProc  
(AVAnnotHandler annotHandler, AVDoc fromDoc, PDAnnot anAnnot,  
AVDoc toDoc);
```

| | |
|------------------------|--|
| Description | (Optional) Callback for AVAnnotHandler . Called upon to copy an annotation, possibly to another document. Annotation handlers should provide this callback to allow for copying their annotations. Called by AVDocCopyAnnot . |
| Parameters | <i>annotHandler</i> The annotation handler responsible for this annotation. <i>fromDoc</i> The document whose annotation is copied. <i>anAnnot</i> The annotation to copy. <i>toDoc</i> The document to which the annotation is copied. |
| Return Value | The newly-created PDAnnot copy. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocCopyAnnot |

AVAnnotHandlerCursorEnterProc

```
ACCB1 void ACCB2 AVAnnotHandlerCursorEnterProc  
    (AVAnnotHandler annotHandler, PDAnnot anAnnot,  
    AVPageView pageView);
```

Description *(Optional)* Callback for [AVAnnotHandler](#). Called whenever the cursor moves over an annotation handled by this annotation handler.

Parameters

annotHandler
The annotation handler responsible for this annotation.

anAnnot
The annotation.

pageView
The *AVPageView* in which the annotation is located.

Return Value None

Header File *AVExpT.h*

Related Callbacks [AVAnnotHandlerCursorExitProc](#)

AVAnnotHandlerCursorExitProc

```
ACCB1 void ACCB2 AVAnnotHandlerCursorExitProc  
    (AVAnnotHandler annotHandler, PDAnnot anAnnot,  
    AVPageView pageView);
```

| | |
|--------------------------|--|
| Description | <i>(Optional)</i> Callback for AVAnnotHandler . Called whenever the cursor moves off an annotation handled by this annotation handler. |
| Parameters | <i>annotHandler</i> The annotation handler responsible for this annotation. <i>anAnnot</i> The annotation the cursor is exiting from. <i>pageView</i> The <i>AVPageView</i> in which the annotation is located. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVAnnotHandlerCursorEnterProc |

AVAnnotHandlerDeleteInfoProc

```
ACCB1 void ACCB2 AVAnnotHandlerDeleteInfoProc  
    (AVAnnotHandler avanh, AVAnnotHandlerInfo info);
```

| | |
|--------------------------|---|
| Description | <i>(Optional)</i> Callback for AVAnnotHandler . Deletes information associated with an annotation. |
| Parameters | <i>avanh</i> The annotation handler responsible for this annotation. <i>info</i> Information associated with the annotation. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVAnnotHandlerGetInfoProc PDAnnotHandlerDeleteAnnotInfoProc |
| Related Methods | None |

AVAnnotHandlerDoClickProc

```
ACCB1 ASBool ACCB2 AVAnnotHandlerDoClickProc
(AVAnnotHandler annotHandler, PDAnnot hitAnnot,
AVPageView pageView, ASInt16 xHit, ASInt16 yHit,
ASInt16 flags, ASInt16 clickNo);
```

| | |
|--------------------------|---|
| Description | (<i>Optional</i>) Callback for AVAnnotHandler . It handles both left and right mouse button clicks within the annotation. If <i>NULL</i> , the annotation behaves as if the callback returned <i>false</i> . |
| Parameters | <p><i>annotHandler</i></p> <p>The annotation handler responsible for this annotation.</p> <p><i>hitAnnot</i></p> <p>The annotation in which the mouse was clicked.</p> <p><i>pageView</i></p> <p>The <i>AVPageView</i> in which the annotation is located.</p> <p><i>xHit</i></p> <p>The x-coordinate of the mouse click.</p> <p><i>yHit</i></p> <p>The y-coordinate of the mouse click.</p> <p><i>flags</i></p> <p>Indicates which modifier keys are pressed. Must be an OR of the Modifier Keys values.</p> <p><i>clickNo</i></p> <p>1 if this is a single click, 2 if a double click, 3 if triple click.</p> |
| Return Value | <i>true</i> if the callback handled the mouse click, <i>false</i> if it did not. If the callback does not handle the click, it is passed to any annotation at the same location in lower layers. |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVAnnotHandlerDoKeyDownProc |

AVAnnotHandlerDoKeyDownProc

```
ACCB1 ASBool ACCB2 AVAnnotHandlerDoKeyDownProc
(AVAnnotHandler annotHandler, PDAnnot anAnnot, ASUns16 key,
ASInt16 flags);
```

| | |
|--------------------------|--|
| Description | <p>(<i>Optional</i>) Callback for AVAnnotHandler. It is called to handle key presses in the annotation. If <i>NULL</i>, it is as if the <i>callback</i> returned <i>false</i>.</p> <p>Called when there is a key-down event and the annotation is selected and the active tool doesn't want the event.</p> |
| Parameters | <p><i>annotHandler</i></p> <p>The handler responsible for this annotation.</p> <p><i>anAnnot</i></p> <p>The annotation in which the key press occurred.</p> <p><i>key</i></p> <p>The key that was pressed</p> <p><i>flags</i></p> <p>Indicates which modifier keys are pressed. Must be an OR of the Modifier Keys values.</p> |
| Return Value | <p><i>true</i> if the callback handled the key press, <i>false</i> otherwise.</p> |
| Header File | <p><i>AVExpT.h</i></p> |
| Related Callbacks | <p>None</p> |

AVAnnotHandlerDoPropertiesProc

```
ACCB1 void ACCB2 AVAnnotHandlerDoPropertiesProc  
(AVAnnotHandler annotHandler, PDAnnot anAnnot, AVDoc doc);
```

| | |
|--------------------------|---|
| Description | <p>(Optional) Callback for AVAnnotHandler. It displays whatever user interface it wishes to allow a user to change an annotation's properties. Set it to <i>NULL</i> if the annotation type has no user-specified properties.</p> <p>Called when the user selects the "Properties..." item from the "Edit" menu while an annotation of this type is selected. If <i>NULL</i>, the "Properties" menu item is dimmed when a corresponding object is selected.</p> |
| Parameters | <p><i>annotHandler</i></p> <p>The annotation handler responsible for the annotation.</p> <p><i>anAnnot</i></p> <p>The annotation whose properties are set.</p> <p><i>doc</i></p> <p>The document containing the annotation.</p> |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | None |

AVAnnotHandlerDrawProc

```
ACCB1 void ACCB2 AVAnnotHandlerDrawProc  
(AVAnnotHandler annotHandler, PDAnnot anAnnot,  
AVPageView pageView);
```

| | |
|--------------------------|---|
| Description | (<i>Optional</i>) Callback for AVAnnotHandler . It draws an annotation. Set it to <i>NULL</i> if the annotation handler has no <i>Draw</i> method. |
| Parameters | <i>annotHandler</i> The annotation handler responsible for the annotation. <i>anAnnot</i> The annotation to draw. <i>pageView</i> The <i>AVPageView</i> containing the annotation. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | None |

AVAnnotHandlerEnumProc

```
ACCB1 ASBool ACCB2 AVAnnotHandlerEnumProc  
(AVAnnotHandler annotHandler, void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for AVAppEnumAnnotHandlers . It is called once for each annotation handler currently registered with the Acrobat viewer (see AVAppRegisterAnnotHandler). |
| Parameters | <i>annotHandler</i> The annotation handler. <i>clientData</i> User-supplied data that was passed in the call to AVAppEnumAnnotHandlers . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVAppEnumAnnotHandlers |

AVAnnotHandlerGetAnnotViewBBoxProc

```
ACCB1 void ACCB2 AVAnnotHandlerGetAnnotViewBBoxProc
(AVAnnotHandler annotHandler, AVPageView pageView,
PDAnnot anAnnot, AVRect* bbox);
```

| | |
|--------------------------|--|
| Description | Callback for AVAnnotHandler . It returns the rectangle enclosing the annotation on the screen. |
| Parameters | <p><i>annotHandler</i></p> <p>The annotation handler responsible for the annotation.</p> <p><i>pageView</i></p> <p>The <i>AVPageView</i> in which the annotation is located.</p> <p><i>anAnnot</i></p> <p>The annotation whose bounding box is returned.</p> <p><i>bbox</i></p> <p>(Filled by the callback) The annotation's bounding rectangle.</p> |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVAnnotHandlerPtInAnnotViewBBoxProc |

AVAnnotHandlerGetInfoProc

```
ACCB1 AVAnnotHandlerInfo ACCB2 AVAnnotHandlerGetInfoProc  
(AVAnnotHandler avanh) ;
```

| | |
|--------------------------|---|
| Description | <i>(Optional)</i> Callback for AVAnnotHandler . Gets information associated with an annotation. |
| Parameters | <i>avanh</i> The annotation handler responsible for this annotation. |
| Return Value | Information associated with the annotation. |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVAnnotHandlerDeleteInfoProc |
| Related Methods | AVAppRegisterAnnotHandler |

AVAnnotHandlerGetLayerProc

```
ACCB1 ASFixed ACCB2 AVAnnotHandlerGetLayerProc  
(AVAnnotHandler annotHandler, PDAnnot anAnnot);
```

Description Callback for [AVAnnotHandler](#). It returns the annotation's layer. The layer need not be a constant. For example, the Acrobat viewer's built-in text annotations have a different layer depending on whether they are opened or closed. This ensures that a closed text annotation never appears on top of an open text annotation.

Parameters *annotHandler*
The annotation handler responsible for the annotation.

anAnnot
The annotation whose layer is returned.

Return Value The annotation's layer.

Header File *AVExpT.h*

Related Callbacks None

AVAnnotHandlerGetTypeProc

```
ACCB1 ASAtom ACCB2 AVAnnotHandlerGetTypeProc  
(AVAnnotHandler annotHandler);
```

| | |
|------------------------|--|
| Description | <p>(Required) Callback for AVAnnotHandler. It returns an <i>ASAtom</i> indicating the annotation type for which the handler is responsible. This corresponds to the annotation's <i>Subtype</i> key in the PDF file.</p> <p>This is the method that AVAppGetAnnotHandlerByName uses to find the correct handler.</p> |
| Parameters | <p><i>annotHandler</i></p> <p>The annotation handler whose type is returned.</p> |
| Return Value | The annotation type for which this handler is responsible. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVAppRegisterAnnotHandler |

AVAnnotHandlerNewProc

```
ACCB1 ASBool ACCB2 AVAnnotHandlerNewProc
(AVAnnotHandler annotHandler, PDAnnot anAnnot,
AVPageView pageView);
```

| | |
|--------------------------|--|
| Description | (<i>Unused</i>) Callback for AVAnnotHandler . |
| Parameters | <p><i>annotHandler</i></p> <p>The annotation handler.</p> <p><i>anAnnot</i></p> <p>Annotation to modify.</p> <p><i>pageView</i></p> <p>The <i>AVPageView</i> in which the annotation is located.</p> |
| Return Value | <i>true</i> if the new annotation handler is in a valid initial state for its subclass, <i>false</i> otherwise. |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | None |

AVAnnotHandlerNotifyAnnotAddedToSelectionProc

```
ACCB1 void ACCB2 AVAnnotHandlerNotifyAnnotAddedToSelectionProc
(AVAnnotHandler annotHandler, PDAnnot anAnnot,
AVPageView pageView);
```

| | |
|------------------------|---|
| Description | <p>(<i>Optional</i>) Callback for AVAnnotHandler. It is called when an annotation is added to the selection, and should highlight the annotation. Set it to <i>NULL</i> if omitted.</p> <p>To allow only a single annotation to select at a time, keep a global variable containing the selected annotation and on each invocation of <i>NotifyAnnotAddedToSelection</i> first deselect the current selection, if any (that is, if <i>selectedAnnot</i> is non-<i>NULL</i>, call its AVAnnotHandlerNotifyAnnotRemovedFromSelectionProc select the new annotation, and set <i>selectedAnnot</i>. Of course, <i>RemovedFrom</i> should set <i>selectedAnnot</i> to <i>NULL</i>.</p> |
| Parameters | <p><i>annotHandler</i></p> <p>The annotation handler responsible for the annotation that was added to the selection.</p> <p><i>anAnnot</i></p> <p>The annotation that was added to the selection.</p> <p><i>pageView</i></p> <p>The <i>AVPageView</i> containing the annotation that was added to the selection.</p> |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocSetSelection |

AVAnnotHandlerNotifyAnnotRemovedFromSelectionProc

```
ACCB1 void ACCB2
AVAnnotHandlerNotifyAnnotRemovedFromSelectionProc
(AVAnnotHandler annotHandler, PDAnnot anAnnot,
AVPageView pageView);
```

| | |
|------------------------|--|
| Description | (<i>Optional</i>) Callback for AVAnnotHandler . It is called when an annotation is removed from the selection, and should unhighlight the annotation. Set it to <i>NULL</i> if omitted. |
| Parameters | <p><i>annotHandler</i></p> <p>The annotation handler responsible for the annotation that was removed from the selection.</p> <p><i>anAnnot</i></p> <p>The annotation that was removed from the selection.</p> <p><i>pageView</i></p> <p>The <i>AVPageView</i> in which the annotation appears.</p> |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocClearSelection AVDocSetSelection |

AVAnnotHandlerNotifyDestroyProc

```
ACCB1 void ACCB2 AVAnnotHandlerNotifyDestroyProc  
(AVAnnotHandler annotHandler);
```

| | |
|--------------------------|--|
| Description | Currently unused. |
| Parameters | <i>annotHandler</i> The annotation handler. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | None |

AVAnnotHandlerPtInAnnotViewBBoxProc

```
ACCB1 ASBool ACCB2 AVAnnotHandlerPtInAnnotViewBBoxProc
(AVAnnotHandler annotHandler, AVPageView pageView,
 PDAnnot anAnnot, ASInt16 xHit, ASInt16 yHit);
```

| | |
|------------------------|---|
| Description | <p>Callback for AVAnnotHandler. It is called by AVPageViewInvertQuad to determine whether or not a point is within an annotation. The annotation handler is free to determine what it means for the point to be “in” the annotation. For example, if the annotation appears only as the outline of a circle, the point may be “in” the annotation only when it is near the border of the circle, but not elsewhere within the circle.</p> |
| Parameters | <p><i>annotHandler</i></p> <p>The annotation handler responsible for this annotation.</p> <p><i>pageView</i></p> <p>The <i>AVPageView</i> in which the annotation appears.</p> <p><i>anAnnot</i></p> <p>The annotation being tested.</p> <p><i>xHit</i></p> <p>The x-coordinate of the point to test.</p> <p><i>yHit</i></p> <p>The y-coordinate of the point to test.</p> |
| Return Value | <i>true</i> if the point is in the annotation, <i>false</i> otherwise. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVPageViewInvertQuad |

AVAuxDataPerformProc

```
ACCB1 ASBool ACCB2 AVAuxDataPerformProc (ASAtom auxDataType,  
void* auxData, ASInt32 auxDataLen, AVDoc avDoc);
```

| | |
|------------------------|--|
| Description | <i>(Optional)</i> Callback for AVAuxDataHandler . It is called to process auxiliary data sent to the AVDoc using AVDocSendAuxData . This callback must process the data appropriately for whatever <i>auxDataType</i> is sent. If <i>NULL</i> , default behavior is used. |
| Parameters | <i>auxDataType</i> Specifies the type of <i>auxData</i> . This determines how <i>auxData</i> is interpreted. <i>auxData</i> The auxiliary data. <i>auxDataLen</i> The length of <i>auxData</i> , in bytes. <i>avDoc</i> The document with which the auxiliary data is associated. |
| Return Value | <i>true</i> if the data is acted upon, <i>false</i> if nothing is done. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocSendAuxData |

AVComputeEnabledProc

```
ACCB1 ASBool ACCB2 AVComputeEnabledProc (void* data);
```

| | |
|------------------------|---|
| Description | <p>Callback that is used to determine whether or not a menu item, toolbar button, or tool is enabled. If used for a tool, it is one of the optional callbacks for AVTool.</p> <p>It is called before the menu item or toolbar button is displayed, or before a tool is activated. If it returns <i>false</i>, the menu item, toolbar button, or tool is disabled; otherwise it is enabled. If this callback is <i>NULL</i>, the menu item, toolbar button, or tool is always enabled.</p> <p>Each menu item, toolbar button, or tool can have its own <i>AVComputeEnabledProc</i>, or they can be shared.</p> |
| Parameters | <p><i>data</i></p> <p>User-supplied data that was passed in the call to AVMenuItemSetComputeEnabledProc or AVToolButtonSetComputeEnabledProc.</p> |
| Return Value | <p><i>true</i> if the menu item, toolbar button, or tool is enabled, <i>false</i> otherwise.</p> |
| Header File | <p><i>AVExpT.h</i></p> |
| Related Methods | <p>AVMenuItemSetComputeEnabledProc AVToolButtonSetComputeEnabledProc</p> |

AVComputeMarkedProc

```
ACCB1 ASBool ACCB2 AVComputeMarkedProc (void* data);
```

| | |
|------------------------|---|
| Description | <p>Callback that is used to determine whether or not a menu item or toolbar button is marked (a marked menu item has a checkmark next to it, and a marked toolbar button appears selected). It is called before the menu item or toolbar button is displayed. If it returns <i>false</i>, the menu item or toolbar button is not marked, otherwise it is marked.</p> <p>Each menu item and toolbar button can have its own <i>AVComputeMarkedProc</i>, or they can be shared.</p> |
| Parameters | <p><i>data</i></p> <p>User-supplied data that was passed in the call to AVMenuItemSetComputeMarkedProc or AVToolButtonSetComputeMarkedProc.</p> |
| Return Value | <p><i>true</i> if the menu item or toolbar button is marked, <i>false</i> otherwise.</p> |
| Header File | <p><i>AVExpT.h</i></p> |
| Related Methods | <p>AVMenuItemSetComputeMarkedProc AVToolButtonSetComputeMarkedProc</p> |

AVDocEnumProc

```
ACCB1 ASBool ACCB2 AVDocEnumProc (AVDoc doc, void* clientData);
```

| | |
|------------------------|---|
| Description | Callback used by AVAppEnumDocs . It is called once for each open <i>AVDoc</i> . |
| Parameters | <i>doc</i> The current document. Do not close this <i>AVDoc</i> in this callback function. <i>clientData</i> User-supplied data that was passed in the call to AVAppEnumDocs . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVAppEnumDocs |

AVDocSelectionAddedToSelectionProc

```
ACCB1 void* ACCB2 AVDocSelectionAddedToSelectionProc
    (AVDoc doc, void* curData, void* addData, ASBool highlight);
```

| | |
|--------------------------|---|
| Description | Callback for AVDocSelectionServer . Adds the specified item to the selection, highlights it, and returns the new selection containing the newly-added item. |
| Parameters | <p><i>doc</i></p> <p>The document containing the data to add to the selection.</p> <p><i>curData</i></p> <p>Data representing the current selection. Its format is specific to the selection server.</p> <p><i>addData</i></p> <p>Item to add to the selection.</p> <p><i>highlight</i></p> <p><i>true</i> if the selection should be highlighted (because it has not already been highlighted), <i>false</i> if the selection should not be highlighted (because it has already been highlighted by whoever called this callback). See AVDocSetSelection for additional information on highlighting.</p> |
| Return Value | New selection data containing all current selections (that is, the previous selection plus the newly-added selection), or <i>NULL</i> if failure. If the selection server allows only a single item to be selected at a time, clear the previous selection, highlight the selection specified by <i>addData</i> (if <i>highlight</i> is <i>true</i>), and simply return <i>addData</i> . |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVDocSelectionRemovedFromSelectionProc |

AVDocSelectionCanCopyProc

```
ACCB1 ASBool ACCB2 AVDocSelectionCanCopyProc (AVDoc doc,  
void* selData);
```

| | |
|--------------------------|--|
| Description | <p>Callback for AVDocSelectionServer. It is used to determine whether or not the current selection can be copied. This controls, for example, whether or not the Copy menu item is enabled.</p> <p>The Copy menu item is only enabled if the selection server's <i>AVDocSelectionCanCopyProc</i> returns <i>true</i> and the selection server has an AVDocSelectionCopyProc.</p> |
| Parameters | <p><i>doc</i></p> <p>The document containing the selection.</p> <p><i>selData</i></p> <p>The current selection data.</p> |
| Return Value | <i>true</i> if the current selection can be copied, <i>false</i> otherwise. |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVDocSelectionCopyProc |
| Related Methods | AVDocCopySelection |

AVDocSelectionCanCutProc

```
ACCB1 ASBool ACCB2 AVDocSelectionCanCutProc (AVDoc doc,  
void* data);
```

Description

Callback for [AVDocSelectionServer](#). It is used to determine whether or not the current selection can be cut. This controls, for example, whether or not the Cut menu item is enabled.

The Cut menu item is only enabled if the selection server's *AVDocSelectionCanCutProc* returns *true* and the selection server has an [AVDocSelectionCutProc](#).

Parameters

doc

The document containing the current selection.

data

The current selection data.

Return Value

true if the current selection can be cut, *false* otherwise.

Header File

AVExpT.h

Related Callbacks

[AVDocSelectionCutProc](#)

[AVDocSelectionCanPasteProc](#)

AVDocSelectionCanDeleteProc

```
ACCB1 ASBool ACCB2 AVDocSelectionCanDeleteProc (AVDoc doc,  
void* selData);
```

| | |
|--------------------------|---|
| Description | <p>Callback for AVDocSelectionServer. It is used to determine whether or not the current selection can be deleted. This controls, for example, whether or not the Delete menu item is enabled.</p> <p>The Delete menu item is only enabled if the selection server's <i>AVDocSelectionCanDeleteProc</i> returns <i>true</i> and the selection server has an AVDocSelectionDeleteProc.</p> |
| Parameters | <p><i>doc</i></p> <p>The document containing the current selection.</p> <p><i>selData</i></p> <p>The current selection data.</p> |
| Return Value | <i>true</i> if the current selection can be deleted, <i>false</i> otherwise. |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVDocSelectionCanDeleteProc |
| Related Methods | AVDocDeleteSelection |

AVDocSelectionCanPasteProc

```
ACCB1 ASBool ACCB2 AVDocSelectionCanPasteProc (AVDoc doc);
```

| | |
|--------------------------|--|
| Description | <p>Callback for AVDocSelectionServer. It is used to determine whether or not the current selection can be pasted. This controls, for example, whether or not the Paste menu item is enabled.</p> <p>The Paste menu item is only enabled if the selection server's <i>AVDocSelectionCanPasteProc</i> returns <i>true</i> and the selection server has an AVDocSelectionPasteProc.</p> |
| Parameters | <p><i>doc</i></p> <p>The document into which the selection is pasted.</p> |
| Return Value | <p><i>true</i> if the data currently on the clipboard can be pasted, <i>false</i> otherwise.</p> |
| Header File | <p><i>AVExpT.h</i></p> |
| Related Callbacks | <p>AVDocSelectionPasteProc AVDocSelectionCanCutProc</p> |

AVDocSelectionCanPropertiesProc

```
ACCB1 ASBool ACCB2 AVDocSelectionCanPropertiesProc (AVDoc doc,  
void* selData);
```

| | |
|------------------------|--|
| Description | <p>Callback for AVDocSelectionServer. It is used to determine whether or not the current selection has user-specified properties. This controls whether or not the “Properties...” menu item is enabled.</p> <p>The “Properties...” menu item will not be enabled if the selection server does not have a AVDocSelectionPropertiesProc callback.</p> |
| Parameters | <p><i>doc</i></p> <p>The document containing the current selection.</p> <p><i>selData</i></p> <p>The current selection data.</p> |
| Return Value | <p><i>true</i> if the current selection has a Properties UI, <i>false</i> otherwise.</p> |
| Header File | <p><i>AVExpT.h</i></p> |
| Related Methods | <p>AVDocDoSaveAsWithParams</p> |

AVDocSelectionCanSelectAllProc

```
ACCB1 ASBool ACCB2 AVDocSelectionCanSelectAllProc (AVDoc doc,  
void* selData);
```

| | |
|--------------------------|---|
| Description | Callback for AVDocSelectionServer . It is used to determine whether or not the current selection type can perform a “select all” operation. This controls whether or not the Select All menu item is enabled. |
| Parameters | <i>doc</i> The document containing the current selection. <i>selData</i> The current selection’s data. |
| Return Value | <i>true</i> if “select all” can be performed on the current selection type, <i>false</i> otherwise. |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVDocSelectionSelectAllProc |

AVDocSelectionCopyProc

```
ACCB1 ASBool ACCB2 AVDocSelectionCopyProc (AVDoc doc,  
void* selData);
```

| | |
|------------------------|---|
| Description | Callback for AVDocSelectionServer . It copies the selected item to the clipboard. The Acrobat viewer will have already cleared the clipboard and placed some private data onto it so that it can identify the selection server that put data onto the clipboard. Because of this, a plug-in must not clear the clipboard, but only add its private data. In addition, if the current selection can reasonably be represented as text, plug-ins are strongly encouraged to place a text representation of the selection onto the clipboard, in addition to their own private format. |
| Parameters | <p><i>doc</i></p> <p>The document whose selection is copied.</p> <p><i>selData</i></p> <p>The current selection data in <i>doc</i>.</p> |
| Return Value | <i>true</i> if the data was actually copied, <i>false</i> otherwise. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocCopySelection UnixAppClipboardGetItemId |

AVDocSelectionCutProc

```
ACCB1 ASBool ACCB2 AVDocSelectionCutProc (AVDoc doc,  
void* data);
```

| | |
|------------------------|---|
| Description | Callback for AVDocSelectionServer . Cuts the current selection. See the discussion under AVDocSelectionCopyProc for information on how the selection server must use the clipboard. |
| Parameters | <i>doc</i> Document whose selection is cut. <i>data</i> The current selection data in <i>doc</i> . |
| Return Value | <i>true</i> if the data was actually cut, <i>false</i> otherwise. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | UnixAppClipboardGetItemId |

AVDocSelectionDeleteProc

```
ACCB1 ASBool ACCB2 AVDocSelectionDeleteProc (AVDoc doc,  
void* selData);
```

| | |
|------------------------|---|
| Description | Callback for AVDocSelectionServer . Deletes the current selection. |
| Parameters | <i>doc</i> Document whose selection is deleted. <i>selData</i> The current selection in <i>doc</i> . |
| Return Value | <i>true</i> if the data was actually deleted, <i>false</i> otherwise. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocDeleteSelection |

AVDocSelectionEnumPageRangesProc

```
ACCB1 void ACCB2 AVDocSelectionEnumPageRangesProc (AVDoc doc,
    void* selectionData, AVSelectionPageRangeEnumProc enumProc,
    void* clientData);
```

| | |
|--------------------------|--|
| Description | Callback for AVDocSelectionServer . It allows enumeration of the set of pages the selection covers. |
| Parameters | <p><i>doc</i></p> <p>The document containing the selection.</p> <p><i>selectionData</i></p> <p>The current selection data. Its content and organization is up to the selection server for the current selection type.</p> <p><i>enumProc</i></p> <p>The current selection data. Its content and organization is up to the selection server for the current selection type.</p> <p><i>clientData</i></p> <p>User-supplied data that was passed in the call to AVDocSelectionEnumPageRanges.</p> |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVDocSelectionEnumSelectionProc |

AVDocSelectionEnumSelectionProc

```
ACCB1 void ACCB2 AVDocSelectionEnumSelectionProc (AVDoc doc,
void* data, AVSelectionEnumProc proc, void* clientData);
```

| | |
|--------------------------|---|
| Description | <p>(Optional) Callback for AVDocSelectionServer. Called by AVDocEnumSelection. This callback enumerates the current selection, calling the specified AVSelectionEnumProc for each “item” in the selection (the selection server is free to decide what constitutes an item).</p> <p>If omitted, the selection is enumerated by calling <i>proc</i> once, passing the entire selection to it.</p> |
| Parameters | <p><i>doc</i></p> <p>The document whose selection is enumerated.</p> <p><i>data</i></p> <p>The current selection in <i>doc</i>.</p> <p><i>proc</i></p> <p>The procedure to call for each item in the selection. This callback must halt enumeration if <i>proc</i> returns <i>false</i>, and continue enumeration if <i>proc</i> returns <i>true</i>.</p> <p><i>clientData</i></p> <p>User-supplied data that was passed in the call to AVDocEnumSelection. Pass this as the <i>clientData</i> each time <i>proc</i> is called.</p> |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVDocSelectionEnumPageRangesProc |
| Related Methods | AVDocEnumSelection |

AVDocSelectionGetSelectionTypeProc

```
ACCB1 ASAtom ACCB2 AVDocSelectionGetSelectionTypeProc  
(AVDoc doc, void* data);
```

| | |
|--------------------------|--|
| Description | Callback for AVDocSelectionServer . It provides a way for a single selection server to register different selection types based on the selection data. If this callback is not supplied, the selection type defaults to the return value from AVDocSelectionGetTypeProc . This callback does not affect existing selection servers. |
| Parameters | <i>doc</i> The document containing the selection. <i>data</i> The current selection data. Its content and organization is up to the selection server for the current selection type. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVDocSelectionGetTypeProc |

AVDocSelectionGettingSelectionProc

```
ACCB1 void ACCB2 AVDocSelectionGettingSelectionProc (AVDoc doc,  
void* selData, ASBool highlight);
```

Description

Callback for [AVDocSelectionServer](#). It is called when the selection is set (for example, via [AVDocSetSelection](#)).

Along with whatever else this callback chooses to do, it must highlight the specified selection (if requested), using the selection server's [AVDocSelectionHighlightSelectionProc](#) callback.

Parameters

doc

The document containing the selection.

selData

The selection data being added.

highlight

If *true*, highlight the selection, *false* otherwise.

Return Value

None

Header File

AVExpT.h

Related Methods

[AVDocSetSelection](#)

AVDocSelectionGetTypeProc

ACCB1 [ASAtom](#) ACCB2 AVDocSelectionGetTypeProc (void);

| | |
|------------------------|---|
| Description | Callback for AVDocSelectionServer . Returns the selection type this server handles (for example, Text or Bookmark). This information is used so that the Acrobat viewer knows which selection server to call. |
| Parameters | None |
| Return Value | The selection type this selection server handles. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocGetSelectionServerByType |

AVDocSelectionHighlightSelectionProc

```
ACCB1 void ACCB2 AVDocSelectionHighlightSelectionProc  
(AVDoc doc, void* data);
```

| | |
|--------------------------|---|
| Description | (Previously known as <i>AVDocHighlightSelectionProc</i>) Callback for AVDocSelectionServer . It highlights the selection. This method is unnecessary if the selection type highlights itself as, for example, annotations do. |
| Parameters | <i>doc</i> The document containing the selection. <i>data</i> The current selection data. Its content and organization is up to the selection server for the current selection type. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVDocSelectionGettingSelectionProc AVDocSelectionLosingSelectionProc |

AVDocSelectionKeyDownProc

```
ACCB1 ASBool ACCB2 AVDocSelectionKeyDownProc (AVDoc doc,  
        void* data, ASUns16 key, ASInt16 flags);
```

| | |
|--------------------------|--|
| Description | (<i>Optional</i>) Callback for AVDocSelectionServer . Handles a key press. Needed only if the selection server processes key presses. |
| Parameters | <p><i>doc</i></p> <p>The document in which the click occurred.</p> <p><i>data</i></p> <p>The current selection data for <i>doc</i>.</p> <p><i>key</i></p> <p>The key that was pressed.</p> <p><i>flags</i></p> <p>Modifier keys that were pressed with key. Must be an OR of the Modifier Keys values.</p> |
| Return Value | <i>true</i> if it the keypress was handled, <i>false</i> if it was not and therefore needs to be passed to the next procedure in the key handling chain. |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | None |

AVDocSelectionLosingSelectionProc

```
ACCB1 void ACCB2 AVDocSelectionLosingSelectionProc (AVDoc doc,
void* selData, ASBool highlight);
```

Description

Callback for [AVDocSelectionServer](#). This method is called by (among others) [AVDocClearSelection](#), to let the selection server responsible for the old selection do whatever cleanup it needs.

Along with whatever else this callback chooses to do, it must unhighlight the specified selection (if requested), using the selection server's [AVDocSelectionHighlightSelectionProc](#) callback.

Parameters

doc

The document whose selection is being cleared.

selData

The current selection data.

highlight

If *true*, the selection specified by *selData* should be unhighlighted, *false* otherwise.

Return Value

None

Header File

AVExpT.h

Related Methods

[AVDocClearSelection](#)

AVDocSelectionPasteProc

```
ACCB1 void ACCB2 AVDocSelectionPasteProc (AVDoc doc);
```

| | |
|--------------------------|--|
| Description | Callback for AVDocSelectionServer . Pastes the current selection from the clipboard. |
| Parameters | <i>doc</i> Document into whose selection the clipboard is pasted. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVDocSelectionCutProc AVDocSelectionCanPasteProc |

AVDocSelectionPropertiesProc

```
ACCB1 void ACCB2 AVDocSelectionPropertiesProc (AVDoc doc,  
void* selData);
```

Description *(Optional)* Callback for [AVDocSelectionServer](#). Displays the “set properties” user interface, if any, for the selection server and lets the user set the server’s properties. This callback is not needed unless the selection server has user-settable properties (for example, text highlight color). This callback is called by [AVDocDoSaveAsWithParams](#).

Parameters

doc
The document in which the selection server’s properties are set.

selData
The current selection data.

Return Value None

Header File *AVExpT.h*

Related Methods [AVDocDoSaveAsWithParams](#)

AVDocSelectionRemovedFromSelectionProc

```
ACCB1 void* ACCB2 AVDocSelectionRemovedFromSelectionProc
    (AVDoc doc, void* curData, void* remData, ASBool highlight);
```

| | |
|--------------------------|---|
| Description | Callback for AVDocSelectionServer . Unhighlights the old item given in <i>remData</i> , and returns a new <i>curData</i> or <i>NULL</i> if failure. |
| Parameters | <p><i>doc</i></p> <p>The document in which an item is removed from the selection.</p> <p><i>curData</i></p> <p>The current selection data.</p> <p><i>remData</i></p> <p>The item to remove from the selection. The content and format of <i>selData</i> differs for each selection server, and is up to the selection server's implementors.</p> <p><i>highlight</i></p> <p>If <i>true</i>, the item removed should be unhighlighted. If <i>false</i>, it should not.</p> |
| Return Value | The new selection data (after the specified item has been removed). |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVDocSelectionAddedToSelectionProc |

AVDocSelectionSelectAllProc

```
ACCB1 void* ACCB2 AVDocSelectionSelectAllProc (AVDoc doc,  
void* selData);
```

| | |
|--------------------------|--|
| Description | Callback for AVDocSelectionServer . Selects all items of the current type. |
| Parameters | <i>doc</i> The document in which the “Select All” is performed. <i>selData</i> The current selection data in <i>doc</i> . |
| Return Value | The new selection data, after all items of the specified type have been selected. |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVDocSelectionCanSelectAllProc |

AVDocSelectionShowSelectionProc

```
ACCB1 void ACCB2 AVDocSelectionShowSelectionProc (AVDoc doc,  
void* data);
```

| | |
|------------------------|---|
| Description | Callback for AVDocSelectionServer . Changes the view (for example, by scrolling the current page or moving to the appropriate page) so that the current selection is visible. |
| Parameters | <i>doc</i> The document whose selection is displayed. <i>data</i> The current selection data in <i>doc</i> . |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocShowSelection |

AVExecuteProc

```
ACCB1 void ACCB2 AVExecuteProc (void* data);
```

Description

Callback that is called whenever a menu item or toolbar button is executed. It implements whatever the menu item or toolbar button does (for example, opening a file or initiating a search).

This method may also be called from an external application displaying a PDF file in its window, using the [ExternalDocServerCreationData](#) structure.

Parameters

data

User-supplied data that was passed when [AVMenuItemSetExecuteProc](#) or [AVToolButtonSetExecuteProc](#) were called.

Return Value

None

Header File

AVExpT.h

Related Methods

[AVMenuItemSetExecuteProc](#)
[AVToolButtonSetExecuteProc](#)

AVIdleProc

```
ACCB1 void ACCB2 AVIdleProc (void* clientData);
```

| | |
|------------------------|--|
| Description | Callback that is called periodically when the Acrobat viewer is otherwise idle. |
| Parameters | <i>clientData</i> User-supplied data that was passed in the call to AVAppRegisterIdleProc . |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVAppRegisterIdleProc AVAppUnregisterIdleProc |

AVMenuPredicate

```
ACCB1 ASBool ACCB2 AVMenuPredicate (AVMenu menu,  
    void* clientData);
```

| | |
|------------------------|---|
| Description | Callback that is called for each menu enumerated by AVMenubarAcquireMenuByPredicate . The first menu for which this callback returns <i>true</i> is acquired. |
| Parameters | <i>menu</i> The current menu in the enumeration. <i>clientData</i> User-supplied data that was passed in the call to AVMenubarAcquireMenuByPredicate . |
| Return Value | <i>true</i> to acquire the current menu and halt enumeration, <i>false</i> to continue enumeration. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVMenubarAcquireMenuByPredicate |

AVMenuItemPredicate

```
ACCB1 ASBool ACCB2 AVMenuItemPredicate (AVMenuItem menuItem,  
void* clientData);
```

| | |
|------------------------|--|
| Description | Callback that is called for each menu item enumerated by AVMenubarAcquireMenuItemByPredicate . The first menu item for which this callback returns <i>true</i> is acquired. |
| Parameters | <i>menuItem</i> The current menu item in the enumeration. <i>clientData</i> User-supplied data that was passed in the call to AVMenubarAcquireMenuItemByPredicate . |
| Return Value | <i>true</i> to acquire the current menu item and halt enumeration, <i>false</i> to continue enumeration. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVMenubarAcquireMenuItemByPredicate |

AVPageViewClickProc

```
ACCB1 ASBool ACCB2 AVPageViewClickProc (AVPageView pageView,
    ASInt16 x, ASInt16 y, ASInt16 flags, ASInt16 clickNo,
    void* data);
```

| | |
|------------------------|--|
| Description | User-supplied callback that is called whenever there is a mouse click in its <i>AVPageView</i> . This callback is registered using AVAppRegisterForPageViewClicks . |
| Parameters | <p><i>pageView</i></p> <p>The <i>AVPageView</i> in which the click occurred.</p> <p><i>x</i></p> <p>The click's x-coordinate.</p> <p><i>y</i></p> <p>The click's y-coordinate.</p> <p><i>flags</i></p> <p>Modifier keys that are held down while the mouse was clicked. They must be an OR of the Modifier Keys value.</p> <p><i>clickNo</i></p> <p>1 if single click, 2 if double-click, 3 if triple-click.</p> <p><i>data</i></p> <p>User-supplied data that was passed in the call to .</p> |
| Return Value | <i>true</i> if the callback handled the mouse click, <i>false</i> if it does not and the click should be passed on to the next click handler. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVAppRegisterForPageViewClicks AVAppUnregisterForPageViewClicks |

AVPageViewCursorProc

```
ACCB1 ASBool ACCB2 AVPageViewCursorProc (AVPageView pageView,  
      ASInt16 x, ASInt16 y, void* data);
```

| | |
|------------------------|---|
| Description | User-supplied callback that is called whenever the cursor's shape is adjusted. This callback is registered using AVAppRegisterForPageViewAdjustCursor . |
| Parameters | <p><i>pageView</i> The <i>AVPageView</i> in which the cursor is located.</p> <p><i>x</i> The cursor's <i>x</i>-coordinate.</p> <p><i>y</i> The cursor's <i>y</i>-coordinate.</p> <p><i>data</i> User-supplied data that was passed in the call to AVAppRegisterForPageViewAdjustCursor.</p> |
| Return Value | <i>true</i> if the callback handled the cursor shape, <i>false</i> if it does not and the cursor handler should be allowed to. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVAppRegisterForPageViewAdjustCursor AVAppUnregisterForPageViewAdjustCursor |

AVPageViewDrawProc

```
ACCB1 void ACCB2 AVPageViewDrawProc (AVPageView pageView,  
    AVRect* updateRect, void* data);
```

| | |
|------------------------|---|
| Description | User-supplied callback that is called whenever the <i>AVPageView</i> is drawn. This callback is registered using AVAppRegisterForPageViewDrawing . |
| Parameters | <i>pageView</i> The <i>AVPageView</i> to redraw. <i>updateRect</i> The rectangle enclosing the region to redraw. <i>data</i> User-supplied data that was passed in the call to AVAppRegisterForPageViewDrawing . |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVAppRegisterForPageViewDrawing AVAppUnregisterForPageViewDrawing |

AVSelectionEnumProc

```
ACCB1 ASBool ACCB2 AVSelectionEnumProc (AVDoc doc,  
void* clientData, void* aSelectedObject);
```

| | |
|------------------------|---|
| Description | User-supplied callback that is passed in the call to AVDocEnumSelection . It is called once for each “item” in the selection. AVDocEnumSelection calls the AVDocSelectionEnumSelectionProc for the current selection’s server to actually enumerate the selection. |
| Parameters | <p><i>doc</i></p> <p>The document whose selection is being enumerated.</p> <p><i>clientData</i></p> <p>User-supplied data that was passed in the call to AVDocEnumSelection.</p> <p><i>aSelectedObject</i></p> <p>The selected “item” currently being enumerated. The format of the data is up to the selection server. See Selection Types for a list of the data formats for the Acrobat viewer’s built-in selection servers.</p> |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocEnumSelection |

AVSelectionPageRangeEnumProc

```
ACCB1 ASBool ACCB2 AVSelectionPageRangeEnumProc (AVDoc doc,  
void* clientData, ASInt32 firstPage, ASInt32 lastPage);
```

| | |
|------------------------|---|
| Description | User-supplied callback that is passed in the call to AVDocSelectionEnumPageRanges . It is called once for each page in the selection, and consecutive pages are grouped into a single page range.. |
| Parameters | <p><i>doc</i></p> <p>The document whose selection is being enumerated.</p> <p><i>clientData</i></p> <p>User-supplied data that was passed in the call to AVDocSelectionEnumPageRanges.</p> <p><i>firstPage</i></p> <p>The first page in a consecutive range of pages with a selection.</p> <p><i>lastPage</i></p> <p>The first page in a consecutive range of pages with a selection.</p> |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocEnumSelection |

AVSetCursorProc

```
ACCB1 void ACCB2 AVSetCursorProc (CursHandle curs,  
    void* clientData);
```

| | |
|------------------------|--|
| Description | <i>(Macintosh only)</i> Callback in ExternalDocWindowData . Called for a mouse-related event, such as mouse down or mouse movement. |
| Parameters | <i>curs</i> Handle to cursor. It is a <i>CursHandle</i> . <i>clientData</i> User-supplied data that was passed in ExternalDocWindowData . |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocIsReadOnly AVDocOpenFromFileWithParams AVDocOpenFromPDDocWithParams |

AVSetMessageProc

```
ACCB1 void ACCB2 AVSetMessageProc (char* msg,  
    void* clientData);
```

| | |
|------------------------|---|
| Description | (<i>Unused</i>) Callback in ExternalDocServerCreationData . |
| Parameters | <i>doc</i> Message for external application. <i>clientData</i> User-supplied data that was passed in the call to <i>AVSetMessageProc</i> . |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocIsReadOnly AVDocOpenFromFileWithParams AVDocOpenFromPDDocWithParams |

AVSystemFontEnumProc

```
ACCB1 ASBool ACCB2 AVSystemFontEnumProc  
(AVSystemFont systemFont, void* clientData);
```

| | |
|------------------------|---|
| Description | (<i>Macintosh only</i>). Callback for AVAppEnumSystemFonts . It is called once for each system font. |
| Parameters | <i>systemFont</i> The <i>AVSystemFont</i> currently being enumerated. <i>clientData</i> User-supplied data that was passed in the call to AVAppEnumSystemFonts . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVAppEnumSystemFonts |

AVTextCopyProc

```
ACCB1 void ACCB2 AVTextCopyProc (ASAtom format, void* buf,  
    ASInt32 bufLen, void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for AVDocGetPageText . Text is passed to it in the specified format. |
| Parameters | <p><i>format</i></p> <p>Text format. See the description of the <i>format</i> parameter of AVDocGetPageText for a list of the allowed types.</p> <p><i>buf</i></p> <p>The text.</p> <p><i>bufLen</i></p> <p>Length of <i>buf</i>, in bytes.</p> <p><i>clientData</i></p> <p>User-supplied data that was passed in the call to AVDocGetPageText.</p> |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocGetPageText |

AVToolButtonEnumProc

```
ACCB1 ASBool ACCB2 AVToolButtonEnumProc (AVToolButton button,  
void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for AVToolBarEnumButtons . It is called once for each toolbar button. |
| Parameters | <i>button</i> The toolbar button currently being enumerated. <i>clientData</i> User-supplied data that was passed in the call to AVToolBarEnumButtons . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVToolBarEnumButtons |

AVToolEnumProc

```
ACCB1 ASBool ACCB2 AVToolEnumProc (AVTool tool,  
    void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for AVAppEnumTools . It is called once for each tool. |
| Parameters | <i>tool</i> The tool currently being enumerated. <i>clientData</i> User-supplied data that was passed in the call to AVAppEnumTools . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVAppEnumTools |

AVTransHandlerCompleteTransDictProc

```
ACCB1 void ACCB2 AVTransHandlerCompleteTransDictProc
    (AVTransHandler avth, const char* uiName, CosObj transDict);
```

Description

Callback for [AVTransHandler](#). This method is called after the user has selected a distinct transition. The transition handler must fill in any dictionary items necessary to create the effect specified by the *uiName* passed in. For example, if the “Wipe” transition handler is passed an *uiName* of “Wipe Left,” it would set the **Dir** key in *transDict* to the value 180.

[AVTransHandlerCompleteTransDictProc](#) should fill in standard information like direction, dimension, motion, and so forth—information gathered entirely from the UI name. Other specific informaton should be filled in by [AVTransHandlerDoPropertiesProc](#).

Parameters

avth

The transition handler.

uiName

User interface name of the transition.

transDict

Transition dictionary to set.

Return Value

None

Header File

AVExpT.h

Related Callbacks

[AVTransHandlerDoPropertiesProc](#)

AVTransHandlerDoPropertiesProc

```
ACCB1 void ACCB2 AVTransHandlerDoPropertiesProc
(AVTransHandler avth, const char* uiName, PDTrans trans);
```

Description

Callback for [AVTransHandler](#). This method is called when the user clicks the button in the transition settings dialog. This allows the transition to bring up its own custom dialog allowing the user to further specify the desired transition effect.

Once the user selects a transition effect from the popup menu, the viewer immediately creates a transition (using [PDTransNewFromCosDoc](#) or [PDTransNew](#)) and calls [AVTransHandlerInitTransDictProc](#) and [AVTransHandlerCompleteTransDictProc](#). If the handler provides both an [AVTransHandlerDoPropertiesProc](#) and [AVTransHandlerGetButtonTextProc](#) callback, the dialog box displays a button. When the user clicks on the button, the viewer calls the handler's [AVTransHandlerDoPropertiesProc](#) callback.

DoProperties is responsible for making any needed alterations to the transition; *InitTransDict* and *CompleteTransDict* are not called after *DoProperties*.

After the user clicks "OK" in the dialog box, *trans* is filled in using the supplied data.

Parameters

avth

The transition handler.

uiName

The user-interface name for the transition handled by *avth*.

trans

The *PDTrans* to initialize.

Return Value

None

Header File

AVExpT.h

Related Callbacks

[AVTransHandlerCompleteTransDictProc](#)
[AVTransHandlerInitTransDictProc](#)

AVTransHandlerEnumProc

```
ACCB1 ASBool ACCB2 AVTransHandlerEnumProc (AVTransHandler avth,  
void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for AVAppEnumTransHandlers . It is called once for each transition handler. |
| Parameters | <i>avth</i> The transition handler. <i>clientData</i> User-supplied data that was passed in the call to AVAppEnumTransHandlers . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVAppEnumTransHandlers |

AVTransHandlerExecuteProc

```
ACCB1 void ACCB2 AVTransHandlerExecuteProc (AVTransHandler avth,  
      PDTrans trans, AVTransitionPort srcTP, AVTransitionPort dstTP,  
      ASFixed duration);
```

Description

Callback for [AVTransHandler](#). Executes the specified transition. The transition handler is responsible for copying the pixels specified by *srcTP* to the location specified by *dstTP*. In the process the handler can create any visual effect it desires, as long as the source pixels are eventually copied over the destination pixels in the end.

The handler should do its best to execute the visual effect in the number of seconds specified by *duration*.

The implementation will ensure that the source and destination rectangles are the same size, though their corners may not coincide.

Parameters

avth

The transition handler.

trans

The transition to execute.

srcTP

Source transition port.

dstTP

Destination transition port.

duration

Duration of the transition, in seconds.

Return Value

None

Header File

AVExpT.h

Related Callbacks

None

AVTransHandlerGetButtonTextProc

```
ACCB1 ASInt32 ACCB2 AVTransHandlerGetButtonTextProc  
(AVTransHandler avth, char* buffer, ASInt32 bufLen);
```

Description Callback for [AVTransHandler](#). Obtains a localized string that appears in the button on the transition settings dialog box. If [AVTransHandlerGetButtonTextProc](#) is *NULL* or the string it returns is empty, no button will appear.

Parameters

avth
The transition handler.

buffer
(Filled by the callback) The button text.

bufLen
Length of *buffer*, in bytes.

Return Value The number of characters copied into *buffer*.

Header File *AVExpT.h*

Related Callbacks [AVTransHandlerDoPropertiesProc](#)

AVTransHandlerGetInstructionsProc

```
ACCB1 ASInt32 ACCB2 AVTransHandlerGetInstructionsProc  
(AVTransHandler avth, char* buffer, ASInt32 bufLen);
```

| | |
|--------------------------|--|
| Description | (<i>Unused</i>) Callback for AVTransHandler . |
| Parameters | <i>avth</i> The transition handler. <i>buffer</i> (<i>Filled by the callback</i>) The instruction text. <i>bufLen</i> Length of <i>buffer</i> , in bytes. |
| Return Value | The number of characters copied into <i>buffer</i> . |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVActionGetInstructionsProc |

AVTransHandlerGetStringOneTextProc

```
ACCB1 ASInt32 ACCB2 AVTransHandlerGetStringOneTextProc  
(AVTransHandler avth, char* buffer, ASInt32 bufLen);
```

| | |
|--------------------------|---|
| Description | Callback for AVTransHandler . Obtains a localized string that appears above the button on the transition settings dialog box. |
| Parameters | <p><i>avth</i></p> <p>The transition handler.</p> <p><i>buffer</i></p> <p>(Filled by the callback) The string text appearing above the button.</p> <p><i>bufLen</i></p> <p>Length of <i>buffer</i>, in bytes.</p> |
| Return Value | The number of characters copied into <i>buffer</i> . |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVTransHandlerGetStringTwoTextProc |

AVTransHandlerGetStringTwoTextProc

```
ACCB1 ASInt32 ACCB2 AVTransHandlerGetStringTwoTextProc  
(AVTransHandler avth, char* buffer, ASInt32 bufLen);
```

| | |
|--------------------------|---|
| Description | Callback for AVTransHandler . Obtains a localized string that appears below the button on the transition settings dialog box. |
| Parameters | <p><i>avth</i></p> <p>The transition handler.</p> <p><i>buffer</i></p> <p>(Filled by the callback) The string text appearing below the button.</p> <p><i>bufLen</i></p> <p>Length of <i>buffer</i>, in bytes.</p> |
| Return Value | The number of characters copied into <i>buffer</i> . |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVTransHandlerGetStringOneTextProc |

AVTransHandlerGetTypeProc

```
ACCB1 ASAtom ACCB2 AVTransHandlerGetTypeProc  
(AVTransHandler avth);
```

| | |
|------------------------|---|
| Description | Callback for AVTransHandler . Gets the transition type serviced by this handler. The handler for a given transition is found by comparing the result of PDTransGetSubtype to the value returned by the registered transition handler's AVTransHandlerGetTypeProc callbacks. |
| Parameters | <i>avth</i> The transition handler. |
| Return Value | Type of transition handler, which may be one of the types provided in the Acrobat viewer or a new type registered by a plug-in. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | PDTransGetSubtype |

AVTransHandlerGetItemUINameProc

```
ACCB1 ASInt32 ACCB2 AVTransHandlerGetItemUINameProc
(AVTransHandler avth, ASInt32 item, char* buffer,
ASInt32 bufLen);
```

Description

Callback for [AVTransHandler](#).

A transition handler can handle several distinct transitions. For example, the “Wipe” transition handler can create four distinct effects: wipe left, wipe right, wipe up, and wipe down.

The transition setting dialog box should create a separate user interface entry for each distinct transition. It determines both the number and names of the distinct transition types by repeatedly calling each transition handler’s

[AVTransHandlerGetItemUINameProc](#) callback, starting with an item number of 0 and increasing until [AVTransHandlerGetItemUINameProc](#) returns an empty string.

Thus when the transaction handler is selected from the list, this callback is called. The transition handler should fill in the **Type** and **S** fields.

[AVTransHandlerGetItemUINameProc](#) should fill in any default values. This information is passed into the [AVTransHandlerDoPropertiesProc](#) in the form of a *PDTrans* if that callback exists.

Parameters

avth

The transition handler.

item

The item number.

buffer

(Filled by the callback) The name of the transition in the user interface. This string should be localized.

bufLen

Length of *buffer*, in bytes.

Return Value

The number of characters copied into *buffer*.

Header File

AVExpT.h

Related Callbacks [AVTransHandlerGetUINameProc](#)

AVTransHandlerGetUINameProc

```
ACCB1 ASInt32 ACCB2 AVTransHandlerGetUINameProc
    (AVTransHandler avth, PDTrans trans, char* buffer,
     ASInt32 bufLen);
```

Description

Callback for [AVTransHandler](#). Retrieves the user-interface name for an existing *PDTrans*. For example, if the transition type is "Wipe" and the direction is 180, *AVTransHandlerGetUINameProc* would return "Wipe Left," localized.

A transition handler can handle several distinct transitions. For example, the "Wipe" transition handler can create four distinct effects: wipe left, wipe right, wipe up, and wipe down.

The transition setting dialog box creates a separate user-interface entry for each distinct transition. It determines both the number and names of the distinct transition types by repeatedly calling each transition handler's [AVTransHandlerGetUINameProc](#) callback, starting with an item number of 0 and increasing until the [AVTransHandlerGetUINameProc](#) callback returns an empty string.

The string returned by [AVTransHandlerGetUINameProc](#) should be localized.

The *AVTransHandlerGetUINameProc* is used to enumerate the entire list of supported transition effects that the handler wishes to display in the popup menu (for example, "Wipe Left" for item == 0, "Wipe Right" for item == 1, and so on).

Parameters

avth

The transition handler.

trans

The transition whose name is obtained.

buffer

(Filled by the callback) The user-interface name of the transition.

bufLen

Length of *buffer*, in bytes.

Return Value

The number of characters copied into *buffer*.

Header File *AVExpT.h*

Related Callbacks [AVTransHandlerGetItemUINameProc](#)

AVTransHandlerInitTransDictProc

```
ACCB1 void ACCB2 AVTransHandlerInitTransDictProc
(AVTransHandler avth, CosObj transDict);
```

Description

Callback for [AVTransHandler](#). This method should set default values in the transition dictionary, *transDict*.

As soon as the handler is selected from the list, [AVTransHandlerInitTransDictProc](#) is called. This function should fill in the **Type** and **S** fields of *transDict*.

[AVTransHandlerInitTransDictProc](#) should also fill in any default values. This information is passed to [AVTransHandlerDoPropertiesProc](#) in the form of a [PDTrans](#) if [AVTransHandlerDoPropertiesProc](#) exists.

Normally the **Type** and **S** fields are filled in when the transition is created via [PDTransNewFromCosDoc](#). The implementation then calls [AVTransHandlerInitTransDictProc](#) and [AVTransHandlerCompleteTransDictProc](#) immediately on the newly-created *PDTrans*.

Parameters

avth

The transition handler.

transDict

Transition dictionary to set.

Return Value

None

Header File

AVExpT.h

Related Callbacks

[AVTransHandlerCompleteTransDictProc](#)

AVWindowAdjustCursorProc

```
ACCB1 void ACCB2 AVWindowAdjustCursorProc (AVWindow win,  
      ASInt16 x, ASInt16 y);
```

| | |
|--------------------------|--|
| Description | Callback for AVWindowHandler . Called periodically while the cursor is over the <i>AVWindow</i> (if the window is active). Use this to adjust the cursor's appearance. |
| Parameters | <i>win</i> The window containing the cursor. <i>x</i> The cursor's <i>x</i> -coordinate. <i>y</i> The cursor's <i>y</i> -coordinate. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | None |

AVWindowCanPerformEditOpProc

```
ACCB1 ASBool ACCB2 AVWindowCanPerformEditOpProc (AVWindow win,  
    ASAtom editOp);
```

| | |
|--------------------------|---|
| Description | Callback for AVWindowHandler . Called before showing the <i>Edit</i> menu, to determine whether or not to enable the <i>Edit</i> menu item corresponding to the given <i>ASAtom</i> . |
| Parameters | <i>win</i> The current window. <i>editOp</i> <i>ASAtom</i> specifying the edit operation. Must be an <i>ASAtom</i> corresponding to one the strings: Cut, Copy, Paste, Clear, SelectAll, and Undo. |
| Return Value | <i>true</i> if the specified operation can be performed, <i>false</i> otherwise. |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVWindowPerformEditOpProc |

AVWindowDestroyPlatformThingProc

```
ACCB1 void ACCB2 AVWindowDestroyPlatformThingProc  
    (AVWindow win, void* platformThing);
```

| | |
|--------------------------|---|
| Description | Callback for AVWindowHandler . Called when it's time to dispose of the <i>platformThing</i> for the window passed to AVWindowNewFromPlatformThing . |
| Parameters | <i>win</i> The window. <i>AVRect</i> The platform-specific object (<i>WindowPtr</i> in Mac OS, an <i>HWND</i> in Windows, and a <i>Widget</i> in UNIX) that was used for this window. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | None |

AVWindowDidActivateProc

```
ACCB1 void ACCB2 AVWindowDidActivateProc (AVWindow win);
```

| | |
|--------------------------|--|
| Description | Callback for AVWindowHandler . Called after the window has been activated. The window being activated will not always become the frontmost window. |
| Parameters | <i>win</i> The window being activated. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVWindowWillDeactivateProc |

AVWindowDidBecomeKeyProc

```
ACCB1 void ACCB2 AVWindowDidBecomeKeyProc (AVWindow win);
```

| | |
|--------------------------|--|
| Description | Callback for AVWindowHandler . Called after the window becomes the key window. |
| Parameters | <i>win</i> The window becoming the key window. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVWindowWillResignKeyProc |
| Related Methods | AVWindowSetWantsKey |

AVWindowDidCloseProc

```
ACCB1 void ACCB2 AVWindowDidCloseProc (AVWindow win);
```

| | |
|--------------------------|---|
| Description | Callback for AVWindowHandler . Called immediately after the window has been closed, but before it has been freed. You may want to explicitly destroy the window in this routine. See also AVWindowWillCloseProc . |
| Parameters | <i>win</i> The window that was closed. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVWindowWillCloseProc |
| Related Methods | AVWindowUserClose |

AVWindowDidResizeProc

```
ACCB1 void ACCB2 AVWindowDidResizeProc (AVWindow win,  
    const AVRect* newFrame);
```

| | |
|--------------------------|---|
| Description | Callback for AVWindowHandler . Called after the window has been resized. |
| Parameters | <i>win</i> The window that was resized. <i>AVRect</i> Rectangle specifying the window's new size and location. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | None |

AVWindowDrawProc

```
ACCB1 void ACCB2 AVWindowDrawProc (AVWindow win,  
    const AVRect* rect);
```

| | |
|------------------------|---|
| Description | Callback for AVWindowHandler . Called whenever the window needs to refresh some part of its interior. It should redraw the contents of the specified rectangle. |
| Parameters | <i>win</i> The window whose content is redrawn. <i>rect</i> The region of <i>win</i> to redraw. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVWindowDrawNow |

AVWindowKeyDownProc

```
ACCB1 void ACCB2 AVWindowKeyDownProc (AVWindow win, char key,  
void* platformEvent);
```

| | |
|--------------------------|---|
| Description | Callback for AVWindowHandler . Called to handle keystrokes when this is the key window. |
| Parameters | <i>win</i> The window. <i>key</i> The key that was pressed. <i>platformEvent</i> Platform-specific event record: Macintosh — Pointer to an <i>EventRecord</i> . UNIX — <i>eventPtr</i> . |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVWindowMouseDownProc |

AVWindowMouseDownProc

```
ACCB1 void ACCB2 AVWindowMouseDownProc (AVWindow win,  
    ASInt16 x, ASInt16 y, void* platformEvent);
```

| | |
|--------------------------|---|
| Description | Callback for AVWindowHandler . Mouse clicks in the <i>AVWindow</i> are dispatched through this callback. |
| Parameters | <p><i>win</i></p> <p>The window in which the click occurred.</p> <p><i>x</i></p> <p>The click's x-coordinate.</p> <p><i>y</i></p> <p>The click's x-coordinate.</p> <p><i>platformEvent</i></p> <p>Platform-specific event record:</p> <p>Macintosh — Pointer to an <i>EventRecord</i>.</p> <p>UNIX — <i>eventPtr</i>.</p> |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVWindowKeyDownProc |

AVWindowPerformEditOpProc

```
ACCB1 void ACCB2 AVWindowPerformEditOpProc (AVWindow win,  
      ASAtom editOp);
```

| | |
|--------------------------|---|
| Description | Callback for AVWindowHandler . Called when the user has chosen an <i>Edit</i> menu item, if the corresponding AVWindowCanPerformEditOpProc returned <i>true</i> . |
| Parameters | <i>win</i> The window that the edit menu item is to act on. <i>editOp</i> An <i>ASAtom</i> specifying the edit operation. Must be an <i>ASAtom</i> corresponding to one the strings: Cut, Copy, Paste, Clear, SelectAll, and Undo. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVWindowCanPerformEditOpProc |

AVWindowWillBeResizedProc

```
ACCB1 ASBool ACCB2 AVWindowWillBeResizedProc (AVWindow win,  
        AVRect* newFrame);
```

| | |
|------------------------|---|
| Description | Callback for AVWindowHandler . Called when the window is about to resize. |
| Parameters | <i>win</i> The window to resize. <i>newFrame</i> Rectangle specifying the size to which the window will be resized. This callback may change the new frame size. |
| Return Value | <i>true</i> to permit the resizing, <i>false</i> to abort it. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVWindowSetFrame |

AVWindowWillCloseProc

```
ACCB1 ASBool ACCB2 AVWindowWillCloseProc (AVWindow win,  
      ASBool quitting);
```

| | |
|------------------------|--|
| Description | Callback for AVWindowHandler . The window is about to close. |
| Parameters | <i>win</i> The window to close. <i>quitting</i> If <i>true</i> , the application is trying to quit. |
| Return Value | <i>true</i> if the window should close, <i>false</i> to abort the operation. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVWindowUserClose |

AVWindowWillDeactivateProc

```
ACCB1 void ACCB2 AVWindowWillDeactivateProc (AVWindow win);
```

| | |
|--------------------------|--|
| Description | Callback for AVWindowHandler . Called before the window becomes deactivated or hidden. |
| Parameters | <i>win</i> The window that will be deactivated. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVWindowDidActivateProc |

AVWindowWillResignKeyProc

```
ACCB1 void ACCB2 AVWindowWillResignKeyProc (AVWindow win);
```

| | |
|--------------------------|--|
| Description | Callback for AVWindowHandler . Called before the window ceases to be the key window. |
| Parameters | <i>win</i> The window to resign key status. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | AVWindowDidBecomeKeyProc |

CancelProc

```
ASBool CancelProc (void* clientData);
```

| | |
|--------------------------|---|
| Description | Callback to check for canceling operations. A <i>CancelProc</i> is typically passed to some method that takes a long time to complete. At frequent intervals, the method calls the <i>CancelProc</i> . If it returns <i>true</i> , then the method cancels its operation; if <i>false</i> , it continues. |
| Parameters | <i>clientData</i> User-supplied data that was passed to the <i>CancelProc</i> . |
| Return Value | <i>true</i> if the processing is canceled, <i>false</i> otherwise. |
| Header File | <i>ASexpt.h</i> |
| Related Callbacks | DoCancel PDFLPrintCancelProc |
| Related Methods | AVAppGetCancelProc |

CosDocEnumEOFsProc

```
ACCB1 ASBool ACCB2 CosDocEnumEOFsProc (CosDoc cosDoc,  
    ASInt32 fileOffset, void* clientData);
```

| | |
|--------------------------|--|
| Description | Callback for CosDocEnumEOFs . Called once for each EOF in a file. |
| Parameters | <i>cosDoc</i> The <i>CosDoc</i> in which the EOF is found. <i>fileOffset</i> The offset into the file directly following the "%% EOF ". <i>clientData</i> User-supplied data that was passed in the call to CosDocEnumEOFs . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt the enumeration. |
| Header File | <i>CosExpt.h</i> |
| Related Callbacks | None |
| Related Methods | CosDocEnumEOFs |

CosObjEnumProc

```
ACCB1 ASBool ACCB2 CosObjEnumProc (CosObj obj, CosObj value,  
void* clientData);
```

Description Callback for [CosObjEnum](#) and [CosDocEnumIndirect](#). Called once for each component of a composite Cos object (dictionary, array, and stream).

Parameters

obj

Dictionary — Key.

Array — Array element.

Stream — The stream's dictionary (the whole thing, not one key at a time).

value

Dictionary — Value associated with the Key.

Array — *NULL* Cos object.

Stream — *NULL* Cos object.

For [CosDocEnumIndirect](#), this is always the *NULL* Cos object.

clientData

User-supplied data that was passed in the call to [CosObjEnum](#).

Return Value *true* to continue enumeration, *false* to halt enumeration.

Header File CosExpT.h

Related Methods [CosObjEnum](#)

CosObjOffsetProc

```
ACCB1 void ACCB2 CosObjOffsetProc (CosObj obj,  
    ASUns32 fileOffset, ASUns32 length, void* clientData);
```

| | |
|--------------------------|---|
| Description | Callback for PDDocSaveParams used by PDDocSaveWithParams . Use this to get information about Cos objects of interest while a <i>PDDoc</i> is being saved. |
| Parameters | <p><i>obj</i></p> <p>The <i>CosObj</i> found.</p> <p><i>fileOffset</i></p> <p>The offset of <i>obj</i> into the PDF file.</p> <p><i>length</i></p> <p>length of <i>obj</i>.</p> <p><i>clientData</i></p> <p>Pointer to user-supplied data passed in the <i>offsetProcClientData</i> parameter of the PDDocSaveParams structure.</p> |
| Return Value | None |
| Header File | <i>CosExpt.h</i> |
| Related Callbacks | None |
| Related Methods | PDDocSaveWithParams |

CosObjSetCallbackFlagProc

```
ACCB1 ASBool ACCB2 CosObjSetCallbackFlagProc (CosObj obj,  
ASBool set);
```

| | |
|--------------------------|---|
| Description | Callback in PDDocPreSaveInfo , which is used by the PDDocPreSaveProc callback. Use this callback to set a flag in each <i>CosObj</i> that you care about, so that you will be called back during the <i>PDDoc</i> 's save and given the Cos object's offset and length. After a PDF file is saved, the Cos objects previously obtained are no longer valid. |
| Parameters | <i>obj</i> The <i>CosObj</i> marked. <i>set</i> <i>true</i> to set the flag to be called back during the save, <i>false</i> otherwise. |
| Return Value | None |
| Header File | <i>CosExpt.h</i> |
| Related Callbacks | None |
| Related Methods | PDDocSaveWithParams |

CrossDocLinkProc

```
ACCB1 AVDoc ACCB2 CrossDocLinkProc (ASPathName path,  
    ASFileSys fileSys, AVDocViewDef viewDef, AVDoc srcDoc,  
    void* data);
```

| | |
|------------------------|--|
| Description | Callback in ExternalDocServerCreationData . Called when a cross-document link is clicked in an AVDoc in an external application's window. |
| Parameters | <p><i>path</i></p> <p>Path to document to which the link points.</p> <p><i>fileSys</i></p> <p>The <i>ASFileSys</i> with which to open document.</p> <p><i>viewDef</i></p> <p>The <i>AVDocViewDef</i> with which to open document.</p> <p><i>srcDoc</i></p> <p>The <i>AVDoc</i> that contains the cross-document link.</p> <p><i>data</i></p> <p>User-supplied data that was passed in the call to <i>CrossDocLinkProc</i>.</p> |
| Return Value | The <i>AVDoc</i> for the new document. |
| Header File | <i>AvExpT.h</i> |
| Related Methods | AVDocIsReadOnly AVDocOpenFromFileWithParams AVDocOpenFromPDDocWithParams |

CrossDocLinkWithDestProc

```
ACCB1 AVDoc ACCB2 CrossDocLinkWithDestProc (ASPathName path,  
      ASFileSys fileSys, AVDocViewDef viewDef, AVDestInfo destInfo,  
      AVDoc srcDoc, void* data);
```

| | |
|------------------------|---|
| Description | Callback in ExternalDocServerCreationData . Called when a cross-document link is clicked in an AVDoc in an external application's window. |
| Parameters | <p><i>path</i> Path to document to which the link points.</p> <p><i>fileSys</i> The <i>ASFileSys</i> with which to open document.</p> <p><i>viewDef</i> The <i>AVDocViewDef</i> with which to open document.</p> <p><i>destInfo</i> A destination in a PDF document.</p> <p><i>srcDoc</i> The <i>AVDoc</i> that contains the cross-document link.</p> <p><i>data</i> User-supplied data that was passed in the call to <i>CrossDocLinkProc</i>.</p> |
| Return Value | The <i>AVDoc</i> for the new document. |
| Header File | <i>AvExpT.h</i> |
| Related Methods | AVDocIsReadOnly AVDocOpenFromFileWithParams AVDocOpenFromPDDocWithParams |

DeactivateProcType

```
ACCB1 void ACCB2 DeactivateProcType (AVTool tool);
```

| | |
|--------------------------|---|
| Description | Callback for AVTool . Called when the tool will no longer be the active tool. |
| Parameters | <i>tool</i> The tool to deactivate. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | ActivateProcType |
| Related Methods | AVAppSetActiveTool |

DoCancel

```
ACCB1 ASBool ACCB2 DoCancel (PDPrintClient printClient);
```

| | |
|------------------------------|---|
| Description | <i>(Optional)</i> Callback for PDPrintClient . Called at least once per page to determine if the print job should be cancelled. |
| Parameters | <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | <i>true</i> if the print job should be canceled, <i>false</i> otherwise. |
| Header File | <i>PDPrint.h</i> |
| Related Callbacks | CancelProc PDFLPrintCancelProc |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

DocBegin

```
ACCB1 void ACCB2 DocBegin (const PDPageRange ranges[],  
    ASInt32 numRanges, ASStm stm, const PDResTree docTree,  
    PDPrintClient printClient);
```

Description *(Optional)* Callback for [PDPrintClient](#). Begin processing a document for printing.

Parameters

ranges
Ranges of pages in document to print.

numRanges
Number of page ranges in document to print.

stm
The PostScript print stream.

docTree
Must be *NULL*.

printClient
The [PDPrintClient](#) from which this is invoked.

Return Value None

Header File *PDPrint.h*

Related Methods [PDDocPrintPages](#)
[PDFLPrintDoc](#)
[PDFLPrintPDF](#)

DocEnd

```
ACCB1 void ACCB2 DocEnd (ASStm stm, PDPrintClient printClient);
```

| | |
|------------------------|---|
| Description | <i>(Optional)</i> Callback for PDPrintClient . |
| Parameters | <i>stm</i> The PostScript print stream. <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | None |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

DoClickProcType

```
ACCB1 ASBool ACCB2 DoClickProcType (AVTool tool,  
    AVPageView pageView, ASInt16 xHit, ASInt16 yHit,  
    ASInt16 flags, ASInt16 clickNo);
```

| | |
|--------------------------|--|
| Description | Callback for AVTool . Handles mouse clicks when the tool is active. For Mac OS, this handles button or option-button mouse clicks. For Windows, this handles right or left button mouse clicks. |
| Parameters | <p><i>tool</i></p> <p>The tool.</p> <p><i>pageView</i></p> <p>The <i>AVPageView</i> in which the click occurred.</p> <p><i>xHit</i></p> <p>The click's x-coordinate.</p> <p><i>yHit</i></p> <p>The click's y-coordinate.</p> <p><i>flags</i></p> <p>Modifier keys that were held down while clicking. Must be an OR of the Modifier Keys values.</p> <p><i>clickNo</i></p> <p>1 if single click, 2 if double click, 3 if triple click.</p> |
| Return Value | <i>true</i> if the callback handled the click, <i>false</i> if it did not and the click should be passed to the next click handling procedure. |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | DoKeyDownProcType |

DoLeaveProcType

```
ACCB1 void ACCB2 DoLeaveProcType (AVTool tool,  
    AVPageView pageView);
```

| | |
|--------------------------|--|
| Description | Callback for AVTool . Called when the tool leaves the page view, that is, when the cursor is moved out of the page view. |
| Parameters | <i>tool</i> The tool. <i>pageView</i> The <i>AVPageView</i> that the tool left. |
| Return Value | None |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | None |

DocSetup

```
ACCB1 ASBool ACCB2 DocSetup (const PDPageRange ranges[],
    ASInt32 numRanges, ASStm prologStm,
    PDPrintClient printClient);
```

| | |
|------------------------|--|
| Description | <i>(Optional)</i> Callback for PDPrintClient . Set up a document for printing. |
| Parameters | <i>ranges</i> Ranges of pages in document to print. <i>numRanges</i> Number of page ranges in document to print. <i>prologStm</i> The prolog stream. <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | <i>false</i> if you don't want PDDocPrintPages to emit procsets and other resources. |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

DoKeyDownProcType

```
ACCB1 ASBool ACCB2 DoKeyDownProcType (AVTool tool, ASUns16 key,  
ASInt16 flags);
```

| | |
|--------------------------|---|
| Description | Callback for AVTool . Handles key presses when the tool is active. |
| Parameters | <i>tool</i> The tool. <i>key</i> The key that was pressed. <i>flags</i> Modifier keys that were also pressed. Must be an OR of the Modifier Keys values. |
| Return Value | <i>true</i> if the callback handled the key press, <i>false</i> if it did not and the key press should be passed to the key press handling procedure. |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | DoClickProcType |

EmitPageContents

```
ACCB1 void ACCB2 EmitPageContents (ASStm stm, ASInt32 pageNum,  
    PDPage pdPage, PDPrintClient printClient);
```

Description *(Optional)* Callback for [PDPrintClient](#). Emit the page contents for printing. Broadcast a notification that the page has been printed:

```
NOTIFY(PDDocDidPrintPage)  
    (doc, page, stm, error)
```

There is a default callback for PostScript printing.

Parameters

stm

The PostScript print stream for this page.

pageNum

The number of the page whose contents is emitted.

pdPage

The *PDPage* to print.

printClient

The [PDPrintClient](#) from which this is invoked.

Return Value

None

Header File

PDPrint.h

Related Methods

[PDDocPrintPages](#)
[PDFLPrintDoc](#)
[PDFLPrintPDF](#)

EmitPrologString

```
ACCB1 ASInt32 ACCB2 EmitPrologString (const char* s,  
    ASInt32 nData, PDPrintClient printClient);
```

| | |
|------------------------|---|
| Description | <i>(Optional)</i> Callback for PDPrintClient . Emits prolog string. Defaults to a method that calls FlushString . |
| Parameters | <i>s</i> Pointer to the prolog string to emit. <i>nData</i> Length of s in bytes. <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | If successful, result is greater than zero. If unsuccessful, result is negative. |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

EmitPSFontBegin

```
ACCB1 void ACCB2 EmitPSFontBegin (ASStm stm, PDFont fontP,  
    PDPrintClient printClient);
```

| | |
|------------------------|---|
| Description | <i>(Optional)</i> Callback for PDPrintClient . Called before a font is emitted (outside DSC comments). |
| Parameters | <i>stm</i> The PostScript print stream for the current page. <i>fontP</i> The font to emit. <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | None |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

EmitPSFontEncodingBegin

```
ACCB1 ASBool ACCB2 EmitPSFontEncodingBegin (ASStm stm,  
      PDFont fontP, PDPrintClient printClient);
```

| | |
|------------------------|---|
| Description | <i>(Optional)</i> Callback for PDPrintClient . Called before a font encoding is emitted. |
| Parameters | <i>stm</i> The PostScript print stream for the current page. <i>fontP</i> The font whose encoding is emitted. <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | Must be <i>true</i> . |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

EmitPSFontEncodingEnd

```
ACCB1 void ACCB2 EmitPSFontEncodingEnd (ASStm stm,  
    PDFont fontP, PDPrintClient printClient);
```

| | |
|------------------------|--|
| Description | <i>(Optional)</i> Callback for PDPrintClient . Called after a font encoding is emitted. |
| Parameters | <i>stm</i> The PostScript print stream for the current page. <i>fontP</i> The font whose encoding was emitted. <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | None |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

EmitPSFontEnd

```
ACCB1 void ACCB2 EmitPSFontEnd (ASStm stm, PDFont fontP,  
    PDPrintClient printClient);
```

| | |
|------------------------|---|
| Description | <i>(Optional)</i> Callback for PDPrintClient . Called after a font is emitted (outside DSC comments). |
| Parameters | <i>stm</i> The PostScript print stream for the current page. <i>fontP</i> The font to emit. <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | None |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

EmitPSResourceBegin

```
ACCB1 void ACCB2 EmitPSResourceBegin (ASStm stm,
ASAtom resType, char* resName, CosObj resObj,
PDPrintClient printClient);
```

| | |
|------------------------|--|
| Description | (<i>Optional</i>) Callback for PDPrintClient . Called before a PostScript resource is emitted. |
| Parameters | <p><i>stm</i></p> <p>The PostScript print stream for the current page.</p> <p><i>resType</i></p> <p>An <i>ASAtom</i> representing the name of the resource type, such as ColorSpace or XObject, as defined in the Portable Document Format Reference Manual, Section 7.5, Resources Dictionaries.</p> <p><i>resName</i></p> <p>Name of the resource in the current page.</p> <p><i>resObj</i></p> <p>The resource's Cos object.</p> <p><i>printClient</i></p> <p>The PDPrintClient from which this is invoked.</p> |
| Return Value | None |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

EmitPSResourceEnd

```
ACCB1 void ACCB2 EmitPSResourceEnd (ASStm stm, ASAtom resType,  
char* resName, CosObj resObj, PDPrintClient printClient);
```

| | |
|------------------------|--|
| Description | (<i>Optional</i>) Callback for PDPrintClient . Called after a PostScript resource is emitted. |
| Parameters | <p><i>stm</i></p> <p>The PostScript print stream for the current page.</p> <p><i>resType</i></p> <p>An <i>ASAtom</i> representing the name of the resource type, such as ColorSpace or XObject, as defined in the Portable Document Format Reference Manual, Section 7.5, Resources Dictionaries.</p> <p><i>resName</i></p> <p>Name of the resource in the current page.</p> <p><i>resObj</i></p> <p>The resource's Cos object.</p> <p><i>printClient</i></p> <p>The PDPrintClient from which this is invoked.</p> |
| Return Value | None |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

EndSetup

```
ACCB1 void ACCB2 EndSetup (ASStm stm, PDPrintClient printClient);
```

Description

(Optional) Callback for [PDPrintClient](#). End the setup for printing a document.

One task that might be performed here is to broadcast a notification for each range of pages that will be printed, as in this pseudo code:

```
for i = 0 to numRanges-1 ...
{
    NOTIFY(PDDocWillPrintPages)(doc,
    fromPage, toPage, psLevel, binaryOK);
    for pageNum = ranges[i]->startPage to
    ranges[i]->lastPage ...
}
```

Parameters

stm

The PostScript print stream.

printClient

The [PDPrintClient](#) from which this is invoked.

Return Value

None

Header File

PDPrint.h

Related Methods

[PDDocPrintPages](#)

[PDFLPrintDoc](#)

[PDFLPrintPDF](#)

FlushString

```
ACCB1 ASInt32 ACCB2 FlushString (char* data, ASInt32 nData,  
    PDPrintClient printClient);
```

| | |
|------------------------|--|
| Description | <i>(Required)</i> Callback for PDPrintClient . Called when the <i>ASStm</i> buffer is full. Buffered data should be sent to the printer, driver, file, or other destination. Not used for non-PostScript printing. |
| Parameters | <i>data</i> Pointer to data to send. <i>nData</i> Length of <i>data</i> , in bytes. <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | None |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

GetFontVMUsage

```
ACCB1 ASInt32 ACCB2 GetFontVMUsage (ASStm stm,  
    PDPrintClient printClient);
```

| | |
|------------------------|---|
| Description | <i>(Optional)</i> Callback for PDPrintClient . Asks client what the VM usage is for the font. |
| Parameters | <i>stm</i> The PostScript print stream. <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | VM usage, in bytes. |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

GetSelectionServerProcType

```
ACCB1 AVDocSelectionServer ACCB2 GetSelectionServerProcType  
    (AVTool tool, AVDoc doc);
```

| | |
|--------------------------|--|
| Description | Callback for AVTool . Gets the selection server associated with the tool, if any. |
| Parameters | <i>tool</i> The tool. <i>doc</i> The active <i>AVDoc</i> . |
| Return Value | The selection server associated with this tool. Returns <i>NULL</i> if the tool has no selection server. |
| Header File | <i>AVExpT.h</i> |
| Related Callbacks | None |

GetTypeProcType

```
ACCB1 ASAtom ACCB2 GetTypeProcType (AVTool tool);
```

| | |
|------------------------|--|
| Description | Callback for AVTool . Returns the tool's name. |
| Parameters | <i>tool</i> The tool. |
| Return Value | The <i>ASAtom</i> corresponding to the tool's name. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVAppGetToolByName |

HFTServerDestroyProc

```
ACCB1 void ACCB2 HFTServerDestroyProc (HFTServer hftServer,  
void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for an <i>HFT</i> server. Destroys the specified <i>HFT</i> (for example, by calling HFTServerDestroy). |
| Parameters | <i>hftServer</i> The <i>HFT</i> server associated with this destroy proc. <i>clientData</i> User-supplied data that was passed in the call to HFTServerNew . |
| Return Value | None |
| Header File | <i>ASExpT.h</i> |
| Related Methods | HFTServerNew HFTDestroy |

HFTServerProvideHFTPProc

```
ACCB1 HFT ACCB2 HFTServerProvideHFTPProc (HFTServer hftServer,  
      ASUns32 version, void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for an <i>HFT</i> server. Returns an <i>HFT</i> with the specified version number. If the <i>HFT</i> has not yet been created, create and return it. If the <i>HFT</i> already exists, do not create a new copy of it; simply return the existing copy. |
| Parameters | <p><i>hftServer</i></p> <p>The <i>HFT</i> server associated with this proc.</p> <p><i>version</i></p> <p>The <i>HFT</i> version being requested.</p> <p><i>clientData</i></p> <p>User-supplied data passed in the call to HFTServerNew.</p> |
| Return Value | The requested version of the <i>HFT</i> . |
| Header File | <i>ASExpT.h</i> |
| Related Methods | HFTServerNew |

IsPersistentProcType

```
ACCB1 ASBool ACCB2 IsPersistentProcType (AVTool tool);
```

Description

Callback for [AVTool](#). Indicates whether or not the tool would like to stay active after it has been used, or is a one-shot tool.

The Acrobat viewer does not contain any code to enforce a tool's request to be persistent; it is up to each tool to be a good citizen. For example, if a tool is not persistent, after that tool is used once it should get the previously active tool (using [AVAppGetLastActiveTool](#)) and check whether or not that tool wishes to be persistent (using [AVToolsPersistent](#)). If so, set the active tool to that tool. If not, set the active tool to the default tool (obtained using [AVAppGetDefaultTool](#)).

Parameters

tool

The tool.

Return Value

true if the tool wishes to be persistent, *false* otherwise.

Header File

AVExpT.h

Related Methods

[AVToolsPersistent](#)

NotifyNewPage

```
ACCB1 void ACCB2 NotifyNewPage (ASInt32 pageNum,  
    PDPrintClient printClient);
```

| | |
|------------------------|---|
| Description | <p>(<i>Optional</i>) Callback for PDPrintClient. Called once for each page printed.</p> <p>A client can use this notification to update the page number in the message pane, for example.</p> |
| Parameters | <p><i>pageNum</i></p> <p>The number of the page to print.</p> <p><i>printClient</i></p> <p>The PDPrintClient from which this is invoked.</p> |
| Return Value | None |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

PageBegin

```
ACCB1 void ACCB2 PageBegin (ASStm stm, ASInt32 pageNum,  
    const PResTree pageTree, PDPrintClient printClient);
```

| | |
|------------------------|---|
| Description | <i>(Optional)</i> Callback for PDPrintClient . Called before a page begins printing. |
| Parameters | <i>stm</i> The PostScript print stream for the current page. <i>pageNum</i> The number of the page being printed. <i>pageTree</i> Must be <i>NULL</i> . <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | None |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

PageEnd

```
ACCB1 void ACCB2 PageEnd (ASStm stm, ASInt32 pageNum,  
    ASBool veryLastPage, PDPrintClient printClient);
```

Description *(Optional)* Callback for [PDPrintClient](#). Called after a page has printed. Broadcasts notification that page has printed:

```
NOTIFY(PDDocDidPrintPages)  
    (doc, fromPage, toPage, error);
```

Parameters

stm

The PostScript print stream for the current page.

pageNum

The number of the page just printed.

veryLastPage

true only for the last page of the very last range,
false otherwise.

printClient

The [PDPrintClient](#) from which this is invoked.

Return Value

None

Header File

PDPrint.h

Related Methods

[PDDocPrintPages](#)
[PDFLPrintDoc](#)
[PDFLPrintPDF](#)

PageSetup

```
ACCB1 ASBool ACCB2 PageSetup (ASStm stm, ASInt32 pageNum,  
    PDPrintClient printClient);
```

| | |
|------------------------|---|
| Description | <i>(Optional)</i> Callback for PDPrintClient . Called before a page is printed to set it up. |
| Parameters | <i>stm</i> The PostScript print stream for the current page. <i>pageNum</i> The number of the page to set up. <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | None |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

PDAnnotHandlerDeleteAnnotInfoProc

```
ACCB1 void ACCB2 PDAnnotHandlerDeleteAnnotInfoProc  
(PDAnnotHandler pdanh, PDAnnotInfo info);
```

| | |
|--------------------------|---|
| Description | <i>(Optional)</i> Callback for PDAnnotHandler . Deletes information associated with an annotation. Frees all the memory associated with the annotation information. |
| Parameters | <i>pdanh</i> The annotation handler responsible for this annotation. <i>info</i> Information associated with the annotation. |
| Return Value | None |
| Header File | <i>PDExpt.h</i> |
| Related Callbacks | AVAnnotHandlerDeleteInfoProc PDAnnotHandlerGetAnnotInfoProc |
| Related Methods | PDRegisterAnnotHandler |

PDAnnotHandlerGetAnnotInfoFlagsProc

```
ACCB1 ASUns32 ACCB2 PDAnnotHandlerGetAnnotInfoFlagsProc  
(PDAnnotHandler pdanh, PDAnnot pdan);
```

| | |
|--------------------------|---|
| Description | Callback for PDAnnotHandler . Gets the annotation handler info flags, which indicate the operations allowed with annotations of this type. |
| Parameters | <p><i>pdanh</i></p> <p>The annotation handler responsible for the annotation.</p> <p><i>pdan</i></p> <p>The annotation.</p> |
| Return Value | <p>Operations allowed. Is an OR of the following flags:</p> <ul style="list-style-type: none">• <i>PDAnnotOperationSummarize</i>—OK to summarize annotations• <i>PDAnnotOperationFilter</i>—OK to filter annotations• <i>PDAnnotOperationManager</i>—OK to manage annotations• <i>PDAnnotIgnorePerms</i>—Allow adding this annotation type to a write-protected document |
| Header File | <i>PDExpt.h</i> |
| Related Callbacks | PDAnnotHandlerGetAnnotInfoProc |
| Related Methods | PDRegisterAnnotHandler |

PDAnnotHandlerGetAnnotInfoProc

```
ACCB1 PDAnnotInfo ACCB2 PDAnnotHandlerGetAnnotInfoProc  
(PDAnnotHandler pdanh, PDAnnot pdan, PDPage pdpage);
```

| | |
|--------------------------|--|
| Description | Callback for PDAnnotHandler . Gets the annotation information for an annotation. |
| Parameters | <p><i>pdanh</i></p> <p>The annotation handler responsible for this annotation.</p> <p><i>pdan</i></p> <p>The annotation for which information is obtained.</p> <p><i>pdpage</i></p> <p>The page associated with the annotation for which information is obtained. If the page associated with the annotation is not known, pass <i>NULL</i>.</p> |
| Return Value | Annotation information, described in a <i>PDAnnotInfo</i> structure. |
| Header File | <i>PDExpt.h</i> |
| Related Callbacks | AVAnnotHandlerGetInfoProc PDAnnotHandlerDeleteAnnotInfoProc |
| Related Methods | PDRegisterAnnotHandler |

PDAnnotHandlerGetTypeProc

```
ACCB1 ASAtom ACCB2 PDAnnotHandlerGetTypeProc  
(PDAnnotHandler pdanh) ;
```

| | |
|--------------------------|---|
| Description | Callback for PDAnnotHandler . Gets an <i>ASAtom</i> indicating the annotation type for which the handler is responsible. This corresponds to the annotation's Subtype key in the PDF file. |
| Parameters | <i>pdanh</i> The annotation handler whose type is returned. |
| Return Value | The annotation type for which this handler is responsible. |
| Header File | <i>PDExpt.h</i> |
| Related Callbacks | PDAnnotHandlerGetAnnotInfoProc |
| Related Methods | PDRegisterAnnotHandler |

PDAnnotWillPrintProc

```
ACCB1 ASBool ACCB2 PDAnnotWillPrintProc (PDAnnotHandler pdanh,  
PDAnnot annot);
```

| | |
|--------------------------|--|
| Description | Callback for PDAnnotHandler . This method is called to determine whether or not an annotation is printed or not. |
| Parameters | <p><i>pdanh</i></p> <p>The annotation handler of the annotation type to print.</p> <p><i>annot</i></p> <p>The annotation to print.</p> |
| Return Value | <i>true</i> if the annotation is printed, <i>false</i> if the annotation is not printed. |
| Header File | <i>PDExpt.h</i> |
| Related Callbacks | PDDocWillExportAnnotProc PDDocWillImportAnnotProc |
| Related Methods | PDRegisterAnnotHandler |

PDAuthProc

```
ACCB1 ASBool ACCB2 PDAuthProc (PDDoc pdDoc);
```

Description

Callback used by [PDDocOpen](#). It is called when an encrypted document is being opened to determine whether or not the user is authorized to open the file.

This callback implements whatever authorization strategy you choose and calls the callbacks of the appropriate security handler (the one that was used to secure the document) to obtain and check authorization data.

The *PDAuthProc* must call the security handler's [PDCryptGetAuthDataProc](#) to obtain whatever authorization data is needed (such as a password), then call [PDDocAuthorize](#) (which is mostly a call to the security handler's [PDCryptAuthorizeProc](#)) to determine whether or not this data authorizes access to the file (for example, did the user provide the correct password). The *PDAuthProc* must also free the authorization data by calling the security handler's [PDCryptFreeAuthDataProc](#) (or [ASfree](#), if the handler does not have a [PDCryptFreeAuthDataProc](#).)

For Acrobat 3.0 and earlier, the correct way to obtain the security handler in a *PDAuthProc* is to call [PDDocGetNewCryptHandler](#), relying on the fact that it returns the security handler if the document has no new security handler, and the fact that at the time the file is opened, it cannot yet have a new security handler. (In the future, one or more new methods may be added to make this procedure more straightforward.)

The Acrobat viewer's built-in authorization procedure works as follows:

Call the security handler's [PDCryptAuthorizeProc](#) with *NULL* authorization data to automatically handle the case where no authorization data is needed (for example, the file has a *NULL* password).

If [PDCryptAuthorizeProc](#) returns true
open the file

```
If PDCryptAuthorizeProc returns false then {
  Loop for i = 1 to 3 {
    Call the security handler's
      PDCryptGetAuthDataProc

    If PDCryptGetAuthDataProc returns true {
      Call PDDocAuthorize

      If returns true {
        /* We got authorization */
        Call the security handler's
          PDCryptFreeAuthDataProc
        exit the loop and return from
          PDAuthProc
      }
      Call the security handler's
        PDCryptFreeAuthDataProc
    }
    /* Failed to get authorization after
       three attempts */
    Display a dialog box indicating that
    user is not authorized to open the file.
  }
  return from PDAuthProc
```

Parameters

pdDoc

The *PDDoc* to open.

Return Value

true if the user is authorized to open the document,
false otherwise.

Header File

PDExpT.h

Related Methods

[PDDocAuthorize](#)
[PDDocOpen](#)

PDAuthProcEx

```
ACCB1 ASBool ACCB2 PDAuthProcEx (PDDoc pdDoc,  
    void* clientData);
```

Description

Callback used by [PDDocOpenEx](#). It is called when an encrypted document is being opened, to determine whether or not the user is authorized to open the file.

This callback implements whatever authorization strategy you choose and calls the callbacks of the appropriate security handler (the one that was used to secure the document) to obtain and check authorization data.

The *PDAuthProcEx* should obtain the authorization data (usually a password) and call [PDDocAuthorize](#). [PDDocAuthorize](#) in turn calls the document encryption handler's *Authorize* function, which returns the permissions that the authorization data enables. [PDDocAuthorize](#) adds these permissions to those currently allowed, and returns the new set of allowed permissions.

Parameters

pdDoc

The *PDDoc* to open.

clientData

User-supplied data that was passed in the call to [PDDocOpenEx](#).

Return Value

true if the user is authorized to open the document, *false* otherwise.

Header File

PDExpT.h

Related Methods

[PDDocAuthorize](#)
[PDDocOpen](#)

PDCharProcEnumProc

```
ACCB1 ASBool ACCB2 PDCharProcEnumProc (char* name,  
    PDCharProc obj, void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDFontEnumCharProcs . It is called once for each character in a Type 3 font. |
| Parameters | <i>name</i> The name of the current character. <i>obj</i> Stream Cos object containing the PDF drawing operators that draw the character. <i>clientData</i> User-supplied data that was passed in the call to PDFontEnumCharProcs . |
| Return Value | <i>true</i> to continue enumerating, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDFontEnumCharProcs |

PDFCryptAuthorizeProc

```
ACCB1 PDPerms ACCB2 PDFCryptAuthorizeProc (PDDoc pdDoc,  
void* authData, PDPerms permWanted);
```

Description

Callback for [PDFCryptHandler](#). Called by [PDDocAuthorize](#) when a user tries to set security for an encrypted document and by a [PDAuthProc](#) when a user tries to open a file.

It must decide, based on the contents of the authorization data structure, whether or not the user is permitted to open the file, and what permissions the user has for this file. The authorization data structure is available in making this decision. Alternate implementations may not require authorization data and may, for example, make authorization decisions based on data contained in the security data structure (use [PDFCryptNewSecurityDataProc](#)).

This callback must not obtain the authorization data (for example, by displaying a user interface into which a user can type a password). Obtaining authorization data is handled by the security handler's [PDFCryptGetAuthDataProc](#), which must be called before this callback. Instead, this callback must work with whatever authorization data is passed to it.

It is legitimate for this callback to be called with *NULL* authorization data; the Acrobat viewer's built-in *authProc* does this in order to support authorization methods that do not require authorization data.

When this callback is invoked to determine whether or not a user is permitted to open a file, *permWanted* will be set *pdPermOpen*. In this case, the file's contents will not yet have been decrypted (since this callback is being asked to permit decryption), and some calls must be avoided. For example, a call that causes a page to be parsed will result in an error since the encrypted contents will be parsed. In general, it is safe to obtain information about the presence or absence of things, or the number of things, and to examine any part of a document at the Cos level.

Parameters

pdDoc

The document for which authorized permissions are being requested.

authData

Authorization data. Its format is security handler-specific; each handler can select its own authorization data format.

permWanted

The permissions being requested. Will be either *pdPermOpen* (if the file is being opened) or *pdPermSecure* (if a request is being made to change the document's security settings).

Return Value

The permissions granted based on the *authData*. For opening, the permissions returned usually should be *pdPermOpen* and some or all of *pdPermPrint*, *pdPermEdit*, and *pdPermCopy*. For setting security, permissions returned should be *pdPermAll*. However, if authorization fails, 0 should be returned.

Header File

PDExpT.h

Related Methods

[PDDocOpen](#)
[PDDocAuthorize](#)

PDFCryptFillEncryptDictProc

```
ACCB1 void ACCB2 PDFCryptFillEncryptDictProc (PDDoc pdDoc,  
CosObj encryptDict);
```

Description

Callback for [PDFCryptHandler](#). Called when an encrypted document is saved. Fills the document's Encrypt dictionary with whatever information the security handler wants to store in the document. The *encryptDict* is automatically written to the PDF file (along with the security handler's name) when the document is saved.

Normally this callback is called after [PDFCryptUpdateSecurityDataProc](#). The Security Data structure can be obtained with a call to [PDFCryptNewSecurityDataProc](#), and the *encryptDict* filled based on this data.

Parameters

pdDoc

The document to save.

encryptDict

A dictionary Cos object to fill with whatever information the security handler wants to store in the PDF file.

Unlike all other strings and streams, direct object elements of the *encryptDict* are not encrypted automatically. If you want them to be encrypted, you must encrypt them before inserting them into the dictionary.

Return Value

None

Header File

PDFExpT.h

Related Methods

[PDDocSave](#)

PDFCryptFreeAuthDataProc

```
ACCB1 void ACCB2 PDFCryptFreeAuthDataProc (PDDoc pdDoc,  
void* authData);
```

| | |
|--------------------------|---|
| Description | <i>(Optional)</i> Callback for PDFCryptHandler . Used to free authorization data acquired via PDFCryptNewAuthDataProc . If this callback is omitted, the viewer defaults to freeing the data using ASfree . |
| Parameters | <i>pdDoc</i> The document whose authorization data is freed. <i>authData</i> <i>(Filled by the callback)</i> Pointer to the document's authorization data. |
| Return Value | None |
| Header File | <i>PDFExpT.h</i> |
| Related Callbacks | PDFCryptNewAuthDataProc |
| Related Methods | PDDocGetNewSecurityInfo |

PDcryptFreeCryptDataProc

```
ACCB1 void ACCB2 PDcryptFreeCryptDataProc (PDDoc pdDoc,  
char* cryptData);
```

| | |
|--------------------------|--|
| Description | <i>(Optional)</i> Callback for PDcryptHandler . Used to free authorization data acquired via PDcryptNewCryptDataProc . If this callback is omitted, the viewer defaults to freeing the data using ASfree . |
| Parameters | <i>pdDoc</i> The document whose encryption/decryption data is freed. <i>cryptData</i> <i>(Filled by the callback)</i> Pointer to the document's encryption/decryption data. |
| Return Value | None |
| Header File | <i>PDExpT.h</i> |
| Related Callbacks | PDcryptNewCryptDataProc |
| Related Methods | PDDocGetNewSecurityInfo |

PDCryptFreeSecurityDataProc

```
ACCB1 void ACCB2 PDCryptFreeSecurityDataProc (PDDoc pdDoc,  
void* secData);
```

| | |
|--------------------------|--|
| Description | <i>(Optional)</i> Callback for PDCryptHandler . Used to free security data acquired via PDCryptNewSecurityDataProc . If this callback is omitted, the viewer defaults to freeing the data using ASfree . |
| Parameters | <i>pdDoc</i> The document whose security data is freed. <i>secData</i> <i>(Filled by the callback)</i> Pointer to the document's security data. |
| Return Value | None |
| Header File | <i>PDExpT.h</i> |
| Related Callbacks | PDCryptNewSecurityDataProc |
| Related Methods | PDDocGetNewSecurityInfo |

PDFCryptGetAuthDataProc

```
ACCB1 ASBool ACCB2 PDFCryptGetAuthDataProc (PDDoc pdDoc,  
PDPerms permWanted, void** authDataP);
```

Description

Callback for [PDFCryptHandler](#). This callback is called from a [PDAuthProc](#) when a file is being opened. It is called after [PDFCryptNewSecurityDataProc](#). It must obtain authorization data (for example, ask the user to enter a password) and populates an authorization data structure with the data.

This callback must call the security handler's [PDFCryptNewAuthDataProc](#), if it wishes, to allocate the authorization data structure. Use of an authorization data structure is optional (an implementation may wish to contain authorization data within the security data structure). The authorization data structure is subsequently used by the security handler's [PDFCryptAuthorizeProc](#) to determine whether or not the user is authorized to open the file.

A security handler can specify the standard password dialog by using [AVCryptGetPassword](#). In this case, the *authData* is a *char **.

Parameters

pdDoc

The document to open.

permWanted

Either *pdPermOpen* or *pdPermSecure*. Since this value is also passed to [PDFCryptAuthorizeProc](#), it may not be necessary for this callback to use *permWanted*.

authDataP

Pointer to authorization data structure. Set to *NULL* if not used.

Return Value

true unless the operation should be canceled (for example, if the user cancels a dialog), *false* otherwise.

Header File

PDFExpT.h

Related Callbacks

[PDFCryptNewAuthDataProc](#)

Related Methods

[PDDocOpen](#)

PDFCryptGetSecurityInfoProc

```
ACCB1 void ACCB2 PDFCryptGetSecurityInfoProc (PDDoc pdDoc,
ASUns32* secInfo);
```

| | |
|------------------------|---|
| Description | <p>(<i>Optional</i>) Callback for PDFCryptHandler. Called by PDDocGetNewSecurityInfo. Extracts the security information from the security data structure, and returns the security information.</p> <p>This function is also used after a "Save As..." to reset the permissions according to the current document.</p> <p>A default set of permissions (pdInfoCanPrint pdInfoCanEdit pdInfoCanCopy pdInfoCanEditNotes (see PDFPerms)) is used if this callback is absent.</p> |
| Parameters | <p><i>pdDoc</i></p> <p>The document whose security info is obtained.</p> <p><i>secInfo</i></p> <p>(<i>Filled by the callback</i>) The document's security info. The value must be an OR of the Security Info Flags. All unused bits in must be set to 1.</p> |
| Return Value | None |
| Header File | <i>PDFExpT.h</i> |
| Related Methods | PDDocGetNewSecurityInfo |

PDFCryptNewAuthDataProc

```
ACCB1 void* ACCB2 PDFCryptNewAuthDataProc ( PDDoc pdDoc );
```

Description *(Optional)* Callback for [PDFCryptHandler](#). Creates a new empty authorization data structure. This structure is subsequently filled by [PDFCryptGetAuthDataProc](#), then passed to [PDFCryptAuthorizeProc](#) and eventually to [ASfree](#).

This callback is not called by the Acrobat viewer, but a security handler may use it if it wishes. The Acrobat viewer's standard security handler does not use this method.

Parameters *pdDoc*
The document for which a new authorization data structure is created.

Return Value The newly-created authorization data structure.

Header File *PDFExpT.h*

Related Callbacks [PDFCryptFreeAuthDataProc](#)
[PDFCryptGetAuthDataProc](#)

PDFCryptNewCryptDataProc

```
ACCB1 void ACCB2 PDFCryptNewCryptDataProc (PDDoc pdDoc,  
char** cryptData, ASInt32* cryptDataLen);
```

| | |
|--------------------------|---|
| Description | Callback for PDFCryptHandler . Sets up the key to be passed to initialize the RC4 cipher for encryption and decryption of a PDF file. It is called when an encrypted document is opened or saved. |
| Parameters | <p><i>pdDoc</i></p> <p>The document for which the key is set.</p> <p><i>cryptData</i></p> <p>(Filled by the callback) The key. <i>cryptData</i> must be allocated by ASmalloc because the Acrobat viewer will free it using ASfree.</p> <p><i>cryptDataLen</i></p> <p>(Filled by the callback) The number of bytes in <i>cryptData</i>. Cannot be greater than 5 bytes.</p> |
| Return Value | None |
| Header File | <i>PDExpT.h</i> |
| Related Callbacks | PDFCryptNewAuthDataProc |

PDCCryptNewSecurityDataProc

```
ACCB1 void* ACCB2 PDCCryptNewSecurityDataProc (PDDoc pdDoc,
CosObj encryptDict);
```

Description

(Optional) Callback for [PDCCryptHandler](#). Creates and populates a new structure that contains whatever security-related information the security handler requires (for example, permissions, whether or not the file has an owner password, whether or not the file has a user password, and other data used internally by the security handler).

This callback is called under two circumstances:

- When a document is opened, it will be called with *encryptDict* set to the document's Encrypt dictionary.
- When the user chooses a new encryption method, it will be called without an *encryptDict*. The handler should return a Security Data structure with default values.

If a security handler does not have this callback, the document's *newSecurityData* field is set to *NULL*.

If a file is to be saved, then [PDCCryptUpdateSecurityDataProc](#) will subsequently be called to allow user interface and modification of the contents.

Security data is freed using [PDCCryptFreeSecurityDataProc](#). If [PDCCryptFreeSecurityDataProc](#) is not defined then [ASfree](#) is used

Parameters

pdDoc

The document for which a new security data structure is created.

encryptDict

If *encryptDict* is a dictionary, this callback must initialize the security data so that it corresponds to the dictionary. Otherwise, it must set up default values in the security data structure.

Return Value

The newly-created security data structure.

Header File

PDExpT.h

Related Callbacks

[PDCCryptFreeSecurityDataProc](#)

PDFCryptUpdateSecurityDataProc

```
ACCB1 ASBool ACCB2 PDFCryptUpdateSecurityDataProc (PDDoc pdDoc,  
    ASAtom* cryptHandler, void** secDataP);
```

Description

Callback for [PDFCryptHandler](#). Updates the security data structure (which contains, among other data, flags specifying whether or not users are permitted to print the file, modify it, or copy its contents to the clipboard), usually by bringing up a dialog box. The current security data can be obtained by calling [PDDocGetSecurityData](#).

The Security Data structure should be updated to reflect the encryption parameters that will be used when saving the file. The encryption parameters are transferred to the Encrypt dictionary by a subsequent callback to [PDFCryptFillEncryptDictProc](#).

The security data should be allocated by [ASmalloc](#) or a relative because the Acrobat viewer frees it using [ASfree](#).

Security data is freed using [PDFCryptFreeSecurityDataProc](#). If [PDFCryptFreeSecurityDataProc](#) is not defined then [ASfree](#) is used

The callback can also update the security handler itself. For example, the built-in encryption switches to no encryption if no passwords or permissions are set in the security dialog box. Return *ASAtomNull* if no encryption is used in the saved file.

Parameters

pdDoc

The document whose security data is updated.

cryptHandler

The current security handler for *pdDoc*. Can be modified to change the security handler. Encryption is turned off if *ASAtomNull* is set.

secDataP

(*Required*) Security data structure. Its content and organization is up to the security handler.

Return Value

Return *true* unless the operation should be canceled (for example, the user clicked on the Cancel button).

| | |
|--------------------------|--|
| Header File | <i>PDExpT.h</i> |
| Related Callbacks | <u>PDCryptValidateSecurityDataProc</u> |
| Related Methods | <u>PDDocGetSecurityData</u> |

PDCryptValidateSecurityDataProc

```
ACCB1 void ACCB2 PDCryptValidateSecurityDataProc (PDDoc pdDoc,  
void* secData);
```

| | |
|--------------------------|--|
| Description | <p>(<i>Optional</i>) Callback for PDCryptHandler. Validates the security data structure, which specifies the user's permissions. This callback may modify the security data structure, for example because the user is not authorized to change the security as they requested. A client may have called PDDocNewSecurityData to obtain a new security data structure, then modified it, and then called PDDocSetNewSecurityData to change the document security. This callback should be called before actually setting the document's security data.</p> <p>This callback is not called automatically by the Acrobat viewer. It must be called, if desired, by the security handler's PDCryptUpdateSecurityDataProc.</p> |
| Parameters | <p><i>pdDoc</i></p> <p>The document whose security data is validated.</p> <p><i>secData</i></p> <p>(<i>May be modified by the callback</i>) The document's security data.</p> |
| Return Value | None |
| Header File | <i>PDExpT.h</i> |
| Related Callbacks | PDCryptUpdateSecurityDataProc |
| Related Methods | PDDocNewSecurityData PDDocSetNewSecurityData |

PDDocEnumProc

```
ACCB1 ASBool ACCB2 PDDocEnumProc (PDDoc pdDoc,  
    void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for PDEnumDocs . It is called once for each open <i>PDDoc</i> . |
| Parameters | <i>pdDoc</i> The <i>PDDoc</i> currently being enumerated. <i>clientData</i> User-supplied data that was passed in the call to PDEnumDocs . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDEnumDocs |

PDDocPreSaveProc

```
ACCB1 void ACCB2 PDDocPreSaveProc (PDDoc pdDoc,  
    PDDocPreSaveInfo preSaveInfo, void* clientData);
```

| | |
|--------------------------|---|
| Description | Callback in the PDDocSaveParams structure used by PDDocSaveWithParams . Use this callback to flag Cos objects you wish to access while a <i>PDDoc</i> is being saved. |
| Parameters | <p><i>pdDoc</i></p> <p>The <i>PDDoc</i> to be saved.</p> <p><i>preSaveInfo</i></p> <p>A PDDocPreSaveInfo structure containing information to use during processing before the save.</p> <p><i>clientData</i></p> <p>User-supplied data that was specified in <i>preSaveProcClientData</i> of the PDDocSaveParams structure.</p> |
| Return Value | None |
| Header File | <i>PDExpt.h</i> |
| Related Callbacks | None |
| Related Methods | PDDocSaveWithParams |

PDDocWillExportAnnotCallback

```
ACCB1 ASBool ACCB2 PDDocWillExportAnnotCallback (PDDoc doc,
PDPage pdpage, PDAnnot annot, CosObj dict);
```

| | |
|--------------------------|---|
| Description | <p>Callback for PDDocExportNotes. Determines whether an annotation is exported or not.</p> <p><i>Note: This is a different callback than PDDocWillExportAnnotProc.</i></p> |
| Parameters | <p><i>doc</i></p> <p>The document from which annotations may be exported.</p> <p><i>pdPage</i></p> <p>The page from which the annotation may be exported.</p> <p><i>annot</i></p> <p>The annotation that may be exported.</p> <p><i>dict</i></p> <p>Copy of <i>annot</i> in a Cos object.</p> |
| Return Value | <i>true</i> to export <i>annot</i> , <i>false</i> to not export <i>annot</i> . |
| Header File | <i>PDExpt.h</i> |
| Related Callbacks | PDDocWillImportAnnotCallback |
| Related Methods | None |

PDDocWillExportAnnotProc

```
ACCB1 ASBool ACCB2 PDDocWillExportAnnotProc  
  (PDAnnotHandler pdanh, PDAnnot src, PDAnnot dst);
```

| | |
|--------------------------|--|
| Description | Callback for PDAnnotHandler . Determines whether an annotation is exported or not. <i>Note: This is a different callback than PDDocWillImportAnnotCallback.</i> |
| Parameters | <p><i>pdanh</i></p> <p>The annotation handler of the annotation type to export.</p> <p><i>src</i></p> <p>The annotation that may be exported.</p> <p><i>dst</i></p> <p>Copy of <i>src</i>, which is actually exported.</p> |
| Return Value | <i>true</i> to export <i>dst</i> , <i>false</i> to not export <i>dst</i> . |
| Header File | <i>PDExpt.h</i> |
| Related Callbacks | PDAnnotWillPrintProc PDDocWillImportAnnotProc |
| Related Methods | PDRegisterAnnotHandler |

PDDocWillImportAnnotCallback

```
ACCB1 ASBool ACCB2 PDDocWillImportAnnotCallback (PDDoc doc,  
          PDPage pdPage, PDAnnot annot);
```

| | |
|--------------------------|--|
| Description | Callback for PDDocImportCosDocNotes and PDDocImportNotes . Determines whether an annotation will be imported or not. <i>Note: This is a different callback than PDDocWillImportAnnotProc.</i> |
| Parameters | <i>doc</i> The document into which annotations may be imported. <i>pdPage</i> The page in which the annotation may be imported. <i>annot</i> The annotation that may be imported. |
| Return Value | <i>true</i> to import <i>annot</i> , <i>false</i> to not import <i>annot</i> . |
| Header File | <i>PDExpt.h</i> |
| Related Callbacks | PDDocWillExportAnnotCallback |
| Related Methods | None |

PDDocWillImportAnnotProc

```
ACCB1 ASBool ACCB2 PDDocWillImportAnnotProc
(PDAnnotHandler pdanh, PDDoc doc, PDPage pdpage,
PDAnnot annot);
```

| | |
|--------------------------|---|
| Description | <p>Callback for PDAnnotHandler. Determines whether an annotation will be imported or not.</p> <p><i>Note: This is a different callback than PDDocWillImportAnnotCallback.</i></p> |
| Parameters | <p><i>pdanh</i></p> <p>The annotation handler of the annotation type to import.</p> <p><i>doc</i></p> <p>The document into which annotations may be imported.</p> <p><i>pdPage</i></p> <p>The page in which the annotation may be imported.</p> <p><i>annot</i></p> <p>The annotation that may be imported.</p> |
| Return Value | <i>true</i> to import <i>annot</i> , <i>false</i> to not import <i>annot</i> . |
| Header File | <i>PDExpt.h</i> |
| Related Callbacks | PDAnnotWillPrintProc PDDocWillExportAnnotProc |
| Related Methods | PDRegisterAnnotHandler |

PDEAttrEnumProc

```
ACCB1 ASBool ACCB2 PDEAttrEnumProc (void* attrHdrP,  
    ASUns32 refCount, ASUns16 size, void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDEAttrEnumTable . It is called once for each attribute in a table. |
| Parameters | <p><i>attrHdrP</i></p> <p>An opaque pointer to the attribute. The actual attribute type is not specified in this function, since the storage mechanism only knows the size of the attribute, not its type.</p> <p><i>refCount</i></p> <p>Reference count of the attribute.</p> <p><i>size</i></p> <p>Size of <i>attrHdrP</i>, in bytes.</p> <p><i>clientData</i></p> <p>User-supplied data that was specified in the call to PDEAttrEnumTable.</p> |
| Return Value | Return <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PEExpT.h</i> |
| Related Methods | PDEAttrEnumTable |

PDEClipEnumProc

```
ACCB1 ASBool ACCB2 PDEClipEnumProc (PDEElement elem,  
void* clientData);
```

| | |
|--------------------------|--|
| Description | Callback for PDEClipFlattenedEnumElems , which enumerates all of a <i>PDEClip</i> 's <i>PDEElements</i> in a flattened manner. |
| Parameters | <i>elem</i> The <i>PDEElement</i> currently being enumerated. <i>clientData</i> User-supplied data that was passed in the call to PDEClipFlattenedEnumElems . |
| Return Value | If <i>false</i> , enumeration halts. If <i>true</i> , enumeration continues. |
| Header File | <i>PEExpt.h</i> |
| Related Callbacks | None |
| Related Methods | PDEClipFlattenedEnumElems |

PDEElementEnumProc

```
ACCB1 ASBool ACCB2 PDEElementEnumProc (PDEElement element,  
void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDEEnumElements . It is called once for each <i>PDEElement</i> in a page's Contents Stream or Resources dictionary. |
| Parameters | <i>element</i> The <i>PDEElement</i> . <i>clientData</i> User-supplied data that was specified in the call to PDEEnumElements . |
| Return Value | Return <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PEExpT.h</i> |
| Related Methods | PDEEnumElements |

PDEObjectDumpProc

```
ACCB1 void ACCB2 PDEObjectDumpProc (PDEObject obj,  
    char* dumpInfo, void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDELogDump or PDEObjectDump . It is called once for each <i>PDEObject</i> , its children, and their attributes for the specified number of levels. |
| Parameters | <p><i>obj</i></p> <p>The <i>PDEObject</i>.</p> <p><i>dumpInfo</i></p> <p>Contains information about an object. Information fields are delimited by tabs. There are no newline characters in this string.</p> <p><i>clientData</i></p> <p>User-supplied data that was specified in the call to PDELogDump or PDEObjectDump.</p> |
| Return Value | None |
| Header File | <i>PEExpT.h</i> |
| Related Methods | PDELogDump PDEObjectDump |

PDFFileSpecAcquireASPathProc

```
ACCB1 ASPathName ACCB2 PDFFileSpecAcquireASPathProc
(void* fileSpecHandlerObj, PDFFileSpec fileSpec,
ASPathName relativeToThisPath);
```

| | |
|------------------------|--|
| Description | Callback for PDFFileSpecHandler . Acquires the <i>ASPath</i> corresponding to a file specification. |
| Parameters | <p><i>fileSpecHandlerObj</i></p> <p>User-supplied data passed in the call to PDRegisterFileSpecHandler.</p> <p><i>fileSpec</i></p> <p>The <i>PDFFileSpec</i> for which an <i>ASPath</i> is acquired.</p> <p><i>relativeToThisPath</i></p> <p>A pathname relative to which the <i>PDFFileSpec</i> is interpreted. If <i>NULL</i>, <i>fileSpec</i> is assumed to be an absolute, not a relative, path.</p> |
| Return Value | The <i>ASPathName</i> corresponding to the specified path. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDFFileSpecAcquireASPath |

PDFFileSpecNewFromASPathProc

```
ACCB1 PDFFileSpec ACCB2 PDFFileSpecNewFromASPathProc
(void* fileSpecHandlerObj, PDDoc pdDoc, ASPathName path,
ASPathName relativeToThisPath);
```

| | |
|------------------------|---|
| Description | Callback for PDFFileSpecHandler . Creates a file specification from an <i>ASPath</i> . |
| Parameters | <p><i>fileSpecHandlerObj</i></p> <p>User-supplied data passed in the call to PDRegisterFileSpecHandler.</p> <p><i>pdDoc</i></p> <p>The <i>PDDoc</i> in which the file specification is created.</p> <p><i>path</i></p> <p>The <i>ASPathName</i> for which a corresponding file specification is created.</p> <p><i>relativeToThisPath</i></p> <p>A pathname relative to which <i>path</i> is interpreted. If <i>NULL</i>, <i>path</i> is assumed to be an absolute, not a relative, path.</p> |
| Return Value | The file specification corresponding to the specified <i>ASPathName</i> . |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDFFileSpecNewFromASPath |

PDFLPrintCancelProc

```
ACCB1 ASBool ACCB2 PDFLPrintCancelProc (PDDoc pdDoc,  
void* clientData);
```

| | |
|--------------------------|---|
| Description | <p>This is called once per page of a document being printed.</p> <p>In addition to giving the ability to cancel the print job, a developer can use the <i>PDFLPrintCancelProc</i> to return control briefly to an application to handle events, update UI elements, and so on. The Library pauses printing until the return from the <i>PDFLPrintCancelProc</i>, since it is single-threaded.</p> |
| Parameters | <p><i>pdDoc</i></p> <p>The document being printed.</p> <p><i>clientData</i></p> <p>User-supplied data that was passed in the PDFLPrintUserParamsRec with the call to PDFLPrintDoc.</p> |
| Return Value | <i>true</i> to cancel the print job, <i>false</i> otherwise. |
| Header File | <i>PDFLPrint.h</i> |
| Related Callbacks | CancelProc DoCancel |
| Related Data | PDFLPrintUserParamsRec |
| Related Methods | None |

PDFontEnumProc

```
ACCB1 ASBool ACCB2 PDFontEnumProc (PDFont font, char* encName,  
void* clientData);
```

| | |
|------------------------|--|
| Description | Callback used by PDDocEnumFonts and PDDocEnumLoadedFonts . It is called once for each font. |
| Parameters | <i>font</i> The font currently being enumerated. <i>encName</i> Unused, contains an empty string. <i>clientData</i> User-supplied data passed in the call to PDDocEnumFonts or PDDocEnumLoadedFonts . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDDocEnumFonts PDDocEnumLoadedFonts |

PDGetDataProc

```
ACCB1 ASBool ACCB2 PDGetDataProc (char* data, ASUns32 lenData,  
    void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDXObjectGetData . It is passed the <i>XObject</i> 's data. Currently, the <i>XObject</i> 's data is read 1 kB at a time and passed to this callback. |
| Parameters | <i>data</i> Buffer containing the <i>XObject</i> 's data. <i>lenData</i> The amount of data in <i>data</i> , in bytes. <i>clientData</i> User-supplied data that was passed in the call to PDXObjectGetData . |
| Return Value | <i>true</i> to continue reading the <i>XObject</i> 's data, <i>false</i> to halt it. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDXObjectGetData |

PDGraphicEnumCacheDeviceProc

```
ACCB1 ASBool ACCB2 PDGraphicEnumCacheDeviceProc (ASFixed* parms,  
void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for PDGraphicEnumMonitor . Called for every d1 (that is, setcachedevice) operator. |
| Parameters | <i>parms</i> Array of numbers containing the 6 parameters passed to the d1 operator. <i>clientData</i> User-supplied data that was passed in the call to PDPageEnumContents . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPageEnumContents |

PDGraphicEnumCharWidthProc

```
ACCB1 ASBool ACCB2 PDGraphicEnumCharWidthProc  
(ASFixedPoint width, void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for PDGraphicEnumMonitor . Called for every d0 (that is, setcharwidth) operator. |
| Parameters | <i>width</i> Array of numbers containing the two parameters passed to the d0 operator. <i>clientData</i> User-supplied data that was passed in the call to PDPageEnumContents . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPageEnumContents |

PDGraphicEnumImageProc

```
ACCB1 ASBool ACCB2 PDGraphicEnumImageProc (PDInlineImage obj,  
void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDGraphicEnumMonitor . Called for every image operator. |
| Parameters | <i>obj</i> Image data. <i>clientData</i> User-supplied data that was passed in the call to PDPageEnumContents . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPageEnumContents |

PDGraphicEnumPathProc

```
ACCB1 ASBool ACCB2 PDGraphicEnumPathProc (PDPath obj,  
void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for PDGraphicEnumMonitor . Called for every path operator. |
| Parameters | <i>obj</i> The path data. <i>clientData</i> User-supplied data that was passed in the call to PDPageEnumContents . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPageEnumContents |

PDGraphicEnumRestoreProc

```
ACCB1 ASBool ACCB2 PDGraphicEnumRestoreProc (void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for PDGraphicEnumMonitor . Called for every Q (restore) operator. |
| Parameters | <i>clientData</i> User-supplied data that was passed in the call to PDPageEnumContents . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPageEnumContents |

PDGraphicEnumSaveProc

```
ACCB1 ASBool ACCB2 PDGraphicEnumSaveProc (void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for PDGraphicEnumMonitor . Called for every Q (save) operator. |
| Parameters | <i>clientData</i> User-supplied data that was passed in the call to PDPageEnumContents . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPageEnumContents |

PDGraphicEnumTextProc

```
ACCB1 ASBool ACCB2 PDGraphicEnumTextProc (PDText obj,  
void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for PDGraphicEnumMonitor . Called for every text operator. |
| Parameters | <i>obj</i> The text object. <i>clientData</i> User-supplied data that was passed in the call to PDPageEnumContents . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPageEnumContents |

PDGraphicEnumXObjectRefProc

```
ACCB1 ASBool ACCB2 PDGraphicEnumXObjectRefProc (char* name,  
    ASFixedRect* bbox, void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDGraphicEnumMonitor . Called for every <i>XObject</i> (Do) operator. |
| Parameters | <i>name</i> The <i>XObject</i> 's name. <i>bbox</i> The <i>XObject</i> 's bounding box, describing the bounding box of the <i>XObject</i> in user space. This is only the case for top-level <i>XObjects</i> . If a Form <i>XObject</i> refers to another <i>XObject</i> , the second <i>XObject</i> 's bounding box is the "infinity" bounding box. <i>clientData</i> User-supplied data that was passed in the call to PDPageEnumContents . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPageEnumContents |

PDGraphicEnumXObjectRefMatrixProc

```
ACCB1 ASBool ACCB2 PDGraphicEnumXObjectRefMatrixProc  
(ASFixedMatrix* matrix, void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDGraphicEnumMonitor . Obtains the current matrix for the subsequent <i>XObject</i> . Called immediately before PDGraphicEnumXObjectRefProc . |
| Parameters | <i>matrix</i> (Filled by the callback) The current transformation matrix for the subsequent <i>XObject</i> whose name is obtained by PDGraphicEnumXObjectRefProc . <i>clientData</i> User-supplied data that was passed in the call to PDPageEnumContents . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPageEnumContents |

PDLaunchActionProc

```
ACCB1 ASBool ACCB2 PDLaunchActionProc  
    (void* fileSpecHandlerObj, PDDoc pdDoc, PDAction pdAction);
```

| | |
|--------------------------|--|
| Description | (<i>Optional</i>) Callback for PDFFileSpecHandler . Launches a specified file. Called when the Acrobat viewer encounters a Launch (GoTo File) action. If this callback is <i>NULL</i> , no launch action is performed. |
| Parameters | <i>fileSpecHandlerObj</i> The registered PDFFileSpecHandler . <i>pdDoc</i> The document containing the Launch action. <i>pdAction</i> The action dictionary. |
| Return Value | <i>true</i> if the handler can do the Launch, <i>false</i> otherwise. |
| Header File | <i>PDExpt.h</i> |
| Related Callbacks | None |

PDPageStmImageDataProc

```
ACCB1 ASBool ACCB2 PDPageStmImageDataProc (ASUns8* data,  
      ASSize_t dataLen, void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for PDPageStmGetInlineImage . Should be called when inline image data is encountered in PDPageStmGetToken . This method may be called multiple times for one inline image. If so, each call provides sequential data for the image. |
| Parameters | <p><i>data</i></p> <p>The image data read so far.</p> <p><i>dataLen</i></p> <p>Length of <i>data</i>, in bytes.</p> <p><i>clientData</i></p> <p>User-supplied data that was passed in the call to PDPageStmGetInlineImage (which may have been passed in the PDPageStmGetToken method).</p> |
| Return Value | <i>true</i> to continue reading the image's data, <i>false</i> to stop reading. |
| Header File | <i>PDExpt.h</i> |
| Related Methods | PDPageStmGetInlineImage PDPageStmGetToken |

PDPPageStmStringOverflowProc

```
ACCB1 void ACCB2 PDPPageStmStringOverflowProc (char* sVal,  
        ASSize_t sValLen, void* clientData);
```

| | |
|------------------------|--|
| Description | Callback used by PDPPageStmGetToken . It is called when the length of a string token exceeds <i>kPDPPageStmStringMax</i> bytes (see <i>PDExpT.h</i>) in PDPPageStmGetToken . |
| Parameters | <i>sVal</i> The string value read so far. <i>sValLen</i> Length of <i>sVal</i> , in bytes. <i>clientData</i> User-supplied data that was passed in the call to PDPPageStmGetToken . |
| Return Value | None |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPPageStmGetToken |

PDPathClosePathProc

```
ACCB1 ASBool ACCB2 PDPathClosePathProc (void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for PDPathEnumMonitor . Called for every path closing operator. |
| Parameters | <i>clientData</i> User-supplied data that was passed in the call to PDPathEnum . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPathEnum |

PDPathCurveToProc

```
ACCB1 ASBool ACCB2 PDPathCurveToProc (ASFixedPoint* p1,  
    ASFixedPoint* p2, ASFixedPoint* p3, void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for PDPathEnumMonitor . Called for every c operator. |
| Parameters | <i>p1, p2, p3</i> The three points needed to specify the curve. <i>clientData</i> User-supplied data that was passed in the call to PDPathEnum . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPathEnum |

PDPathLineToProc

```
ACCB1 ASBool ACCB2 PDPathLineToProc (ASFixedPoint* p1,  
    void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDPathEnumMonitor . Called for every I operator. |
| Parameters | <p><i>p1</i></p> <p>The one point needed to specify the line's ending point.</p> <p><i>clientData</i></p> <p>User-supplied data that was passed in the call to PDPathEnum.</p> |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPathEnum |

PDPathMoveToProc

```
ACCB1 ASBool ACCB2 PDPathMoveToProc (ASFixedPoint* p1,  
    void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDPathEnumMonitor . Called for every m operator. |
| Parameters | <p><i>p1</i></p> <p>The one point needed to specify the location to move to.</p> <p><i>clientData</i></p> <p>User-supplied data that was passed in the call to PDPathEnum.</p> |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPathEnum |

PDPathRectProc

```
ACCB1 ASBool ACCB2 PDPathRectProc (ASFixedPoint* p1,  
    ASFixedPoint* p2, void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for PDPathEnumMonitor . Called for every re operator. |
| Parameters | <i>p1, p2</i> The two points needed to specify the rectangle. <i>clientData</i> User-supplied data that was passed in the call to PDPathEnum . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPathEnum |

PDPathVCurveToProc

```
ACCB1 ASBool ACCB2 PDPathVCurveToProc (ASFixedPoint* p1,  
    ASFixedPoint* p2, void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for PDPathEnumMonitor . Called for every v operator. |
| Parameters | <i>p1, p2</i> The two points needed to specify the curve. <i>clientData</i> User-supplied data that was passed in the call to PDPathEnum . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPathEnum |

PDPathYCurveToProc

```
ACCB1 ASBool ACCB2 PDPathYCurveToProc (ASFixedPoint* p1,  
    ASFixedPoint* p2, void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDPathEnumMonitor . Called for every y operator. |
| Parameters | <i>p1, p2</i> The two points needed to specify the curve. <i>clientData</i> User-supplied data passed in the call to PDPathEnum . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPathEnum |

PDPrintCanEmitFontProc

```
ACCB1 ASBool ACCB2 PDPrintCanEmitFontProc (PDFont fontP,  
      PDPrintClient printClient);
```

| | |
|------------------------|---|
| Description | <p>(<i>Optional</i>) Callback for PDPrintClient. Called to determine whether a font can be emitted into the print job. This is used to determine whether a font is a document-included resource. Only used for PostScript printing.</p> <p>If <i>NULL</i>, the default is to assume that any font can be emitted.</p> |
| Parameters | <p><i>fontP</i></p> <p>The font to check.</p> <p><i>printClient</i></p> <p>The PDPrintClient from which this is invoked.</p> |
| Return Value | None |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

PDPrintEmitFontProc

```
ACCB1 ASBool ACCB2 PDPrintEmitFontProc (ASStm stm,  
    PDFont fontP, PDPrintClient printClient);
```

| | |
|------------------------|---|
| Description | (<i>Required</i>) Callback for PDPrintClient . Emits a font. |
| Parameters | <i>stm</i> The PostScript print stream for the current page. <i>fontP</i> Font to emit. <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | None |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

PDPrintEmitPrologResourceProc

```
ACCB1 void ACCB2 PDPrintEmitPrologResourceProc (ASStm stm,  
ASInt32 resID, PDPrintClient printClient);
```

| | |
|------------------------|---|
| Description | <i>(Required)</i> Callback for PDPrintClient . Takes a <i>resID</i> and creates the PostScript prolog. |
| Parameters | <i>stm</i> The PostScript print stream for the current page. <i>resID</i> PostScript procset resource ID. <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | None |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

PDPrintGetFontEncodingMethodProc

```
ACCB1 ASInt32 ACCB2 PDPrintGetFontEncodingMethodProc  
(PDFont fontP, PDPrintClient printClient);
```

| | |
|------------------------|--|
| Description | <i>(Required for PostScript printing)</i> Callback for PDPrintClient . Asks client which encoding method should be used for the font. |
| Parameters | <i>fontP</i> The font whose encoding method is determined. <i>printClient</i> The PDPrintClient from which this is invoked. |
| Return Value | Encoding method, which must be one of the following: <ul style="list-style-type: none">• <i>kPDDoReencode</i> — Use for Type 1 fonts and substituted fonts.• <i>kPDDoNothing</i> — Use for TrueType Windows font or built-in encoding font.• <i>kPDDoXlate</i> — Use for a TrueType custom font or font with Macintosh encoding. |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

PDResourceEnumColorSpaceProc

```
ACCB1 ASBool ACCB2 PDResourceEnumColorSpaceProc (char* name,  
    CosObj colorSpace, void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDResourceEnumMonitor . It is called for color space resources. |
| Parameters | <p><i>name</i></p> <p>Color space name.</p> <p><i>colorSpace</i></p> <p>The name of the color space as it appears in the Resources dictionary.</p> <p><i>clientData</i></p> <p>User-supplied data that was passed in the call to PDFFormEnumResources.</p> |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDFFormEnumResources |

PDResourceEnumFontProc

```
ACCB1 ASBool ACCB2 PDResourceEnumFontProc (PDFont font,  
char* name, void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for PDResourceEnumMonitor . Procedure called for font resources. |
| Parameters | <i>font</i> The font. <i>name</i> The name of the font as it appears in the Resources dictionary. <i>clientData</i> User-supplied data that was passed in the call to PDFFormEnumResources . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDFFormEnumResources |

PDResourceEnumProcSetProc

```
ACCB1 ASBool ACCB2 PDResourceEnumProcSetProc (char* name,  
        void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDResourceEnumMonitor . Procedure called for <i>ProcSet</i> resources. |
| Parameters | <i>name</i> The name of the <i>ProcSet</i> as it appears in the Resources dictionary. <i>clientData</i> User-supplied data that was passed in the call to PDFormEnumResources . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDFormEnumResources |

PDResourceEnumXObjectProc

```
ACCB1 ASBool ACCB2 PDResourceEnumXObjectProc (PDXObject xObject,  
char* name, void* clientData);
```

| | |
|------------------------|---|
| Description | Callback for PDResourceEnumMonitor . Procedure called for <i>XObject</i> resources. |
| Parameters | <i>xObject</i> The <i>XObject</i> . <i>name</i> The name of the <i>XObject</i> as it appears in the Resources dictionary. <i>clientData</i> User-supplied data that was passed in the call to PDFFormEnumResources . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDFFormEnumResources |

PDStringEncodingProc

```
ACCB1 ASBool ACCB2 PDStringEncodingProc (PDFont font, char* string,  
ASInt32 stringLen, ASFixed delta, void* clientData);
```

Description Callback for [PDTextEnum](#). Called once for each string in a text object.

Parameters

font

The font used for *string*.

string

The string. This string may be converted using [PDFontXlateToHost](#) or [PDFontXlateToUCS](#).

stringLen

The number of bytes in *string*.

delta

The difference, in thousandths of an EM, from the end of the previous string to the beginning of the current string. (An EM is a typographic unit of measurement equal to the size of a font. For example, in a 12-point font, an EM is 12 points.) See the description of the **TJ** operator in Section 8.7.5 in the [Portable Document Format Reference Manual](#).

clientData

User-supplied data that was passed in the call to [PDTextEnum](#).

Return Value

true to continue enumeration, *false* to halt enumeration.

Header File

PDExpT.h

Related Methods

[PDTextEnum](#)

PDSysFontEnumProc

```
ACCB1 ASBool ACCB2 PDSysFontEnumProc (PDSysFont sysFont,  
void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDEnumSysFonts . It is called once for each system font. |
| Parameters | <i>sysFont</i> The system font. <i>clientData</i> User-supplied data that was specified in the call to PDEnumSysFonts . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PSFExpT.h</i> |
| Related Methods | PDEnumSysFonts |

PDTextSelectEnumQuadProc

```
ACCB1 ASBool ACCB2 PDTextSelectEnumQuadProc (void* procObj,  
    ASInt32 page, ASFixedQuad* quad);
```

| | |
|------------------------|---|
| Description | Callback for PDTextSelectEnumQuads . Called once for each quad in a text selection. |
| Parameters | <p><i>procObj</i></p> <p>User-supplied data that was passed in the call to PDTextSelectEnumQuads.</p> <p><i>page</i></p> <p>The page on which the text selection is located.</p> <p><i>quad</i></p> <p>The quad being enumerated.</p> |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDTextSelectEnumQuads |

PDTextSelectEnumTextProc

```
ACCB1 ASBool ACCB2 PDTextSelectEnumTextProc (void* procObj,
    PDFont font, ASFixed size, PDColorValue color, char* text,
    ASInt32 textLen);
```

| | |
|------------------------|--|
| Description | Callback for PDTextSelectEnumText . Called once for each text run (text in the same font, size, color, and on the same line) in a text selection. |
| Parameters | <p><i>procObj</i></p> <p>User-supplied data that was passed in the call to PDTextSelectEnumText.</p> <p><i>font</i></p> <p>The text's font.</p> <p><i>size</i></p> <p>The text's size, in points.</p> <p><i>color</i></p> <p>The text's color.</p> <p><i>text</i></p> <p>The text in the current run.</p> <p><i>Note: This string is not necessarily NULL-terminated.</i></p> <p><i>textLen</i></p> <p>The number of bytes in <i>text</i>.</p> |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDTextSelectEnumText |

PDThumbCreationDrawThumbProc

```
ACCB1 void ACCB2 PDThumbCreationDrawThumbProc (PDThumb thumb,  
void* clientData);
```

Description *(Optional)* Callback for [PDThumbCreationServer](#). Called after [PDThumbCreationGetThumbDataProc](#) and after a *PDThumb* has been created. Gives the server a chance to draw the thumbnail image in a status window. May be *NULL*.

Parameters

thumb
The thumbnail image to draw.

clientData
User-supplied data that was passed in the call to [PDDocCreateThumbs](#).

Return Value None

Header File *PDExpT.h*

Related Callbacks [PDThumbCreationGetThumbDataProc](#)

Related Methods [PDDocCreateThumbs](#)

PDThumbCreationGetThumbDataProc

```
ACCB1 ASBool ACCB2 PDThumbCreationGetThumbDataProc
(PDPage page, ASFixed thumbScale, ASInt32 width,
ASInt32 height, void* thumbData, void* clientData);
```

| | |
|---------------------|--|
| Description | (Optional) Callback for PDThumbCreationServer . Called for each page that does not currently contain a thumbnail image. May be <i>NULL</i> . If it is <i>NULL</i> , the thumbnail data is generated by the default thumbnail generator. |
| Parameters | <p><i>page</i></p> <p>The page for which to create a thumbnail image.</p> <p><i>thumbScale</i></p> <p>The scale to map from the page size to the thumbnail size—the thumbnail size is either 1/8 of the page size, or is limited to <i>MAX_THUMBPAGE_WIDTH</i> and <i>MAX_THUMBPAGE_HEIGHT</i>, whichever is smaller.</p> <p><i>width</i></p> <p>The width of the thumbnail image to create.</p> <p><i>height</i></p> <p>The height of the thumbnail image to create.</p> <p><i>thumbData</i></p> <p>A buffer into which the thumbnail data is copied. This buffer has the size:</p> $\text{rowBytes} = (\text{width} * \text{bitsPerPixel} + 7) / 8;$ $\text{size} = \text{rowBytes} * \text{height};$ <p>where <i>bitsPerPixel</i> is specified as <i>numComponents</i> x <i>bitsPerComponent</i>.</p> <p><i>numComponents</i> is dependent upon the color space. For DeviceRGB, <i>numComponents</i> is 3. For an indexed color space, <i>numComponents</i> is 1.</p> <p><i>clientData</i></p> <p>User-supplied data that was passed in the call to PDDocCreateThumbs.</p> |
| Return Value | <i>true</i> to continue thumbnail image creation, <i>false</i> to halt thumbnail image creation. |

Header File *PDExpT.h*

Related Methods [PDDocCreateThumbs](#)

PDThumbCreationNotifyPageProc

```
ACCB1 ASBool ACCB2 PDThumbCreationNotifyPageProc  
    (ASInt32 pageNum, void* clientData);
```

| | |
|------------------------|---|
| Description | <i>(Optional)</i> Callback for PDThumbCreationServer . Called before processing each page. May be <i>NULL</i> . |
| Parameters | <i>pageNum</i> The page for which to create a thumbnail image. <i>clientData</i> User-supplied data that was passed in the call to PDDocCreateThumbs . |
| Return Value | <i>true</i> to continue thumbnail image creation, <i>false</i> to halt thumbnail image creation. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDDocCreateThumbs |

PDWordProc

```
ACCB1 ASBool ACCB2 PDWordProc (PDWordFinder wObj, PDWord wInfo,  
    ASInt32 pgNum, void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDWordFinderEnumWords . Called once for each word. |
| Parameters | <p><i>wObj</i> The word finder.</p> <p><i>wInfo</i> The current word in the enumeration.</p> <p><i>pgNum</i> The page number on which <i>wInfo</i> is located.</p> <p><i>clientData</i> User-supplied data that was passed in the call to PDWordFinderEnumWords.</p> |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDWordFinderEnumWords |

PDXObjectFilterEnumProc

```
ACCB1 ASBool ACCB2 PDXObjectFilterEnumProc (char* filter,  
    CosObj decodeParms, void* clientData);
```

| | |
|------------------------|--|
| Description | Callback for PDXObjectEnumFilters . Called once for each filter that has been applied to an <i>XObject</i> 's data. |
| Parameters | <i>filter</i> The filter's name. <i>decodeParms</i> The dictionary Cos object containing the filter's decode parameters. <i>clientData</i> User-supplied data that was passed in the call to PDXObjectEnumFilters . |
| Return Value | <i>true</i> to continue enumeration, <i>false</i> to halt enumeration. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDXObjectEnumFilters |

PIExportHFTsProcType

```
ACCB1 ASBool ACCB2 PIExportHFTsProcType (void);
```

| | |
|--------------------------|---|
| Description | <p>(<i>Optional</i>) Callback for PIHandshake. This handshaking function is called by the viewer during initialization to allow a plug-in to export an HFT to other plug-ins.</p> <p>If this callback returns <i>false</i>, the PIUnloadProcType callback is called. The plug-in <i>must</i> remove and release <i>all</i> menu items and other user interface elements, HFT's, HFTServers, release any memory or any other resources allocated, and so on, in its PIUnloadProcType callback.</p> |
| Parameters | None |
| Return Value | <i>true</i> if the HFT was exported successfully, <i>false</i> otherwise. |
| Header File | <i>PIVersn.h</i> |
| Related Callbacks | PIHandshake PIImportReplaceAndRegisterProcType |
| Related Methods | None |

PIHandshake

```
ACCB1 ASBool ACCB2 PIHandshake (ASUns32 handshakeVersion,  
    void* handshakeData);
```

| | |
|--------------------------|--|
| Description | <p>This handshaking function is called by the viewer after a plug-in is loaded into memory. The plug-in should fill out <i>handshakeData</i> to tell the viewer its name and how to call it back to export an HFT, import an HFT, perform initialization, and perform any sort of cleanup needed when the plug-in unloads.</p> <p><i>Every plug-in must provide this function.</i></p> |
| Parameters | <p><i>handshakeVersion</i></p> <p>Version of the <i>handshakeData</i> structure.</p> <p><i>handshakeData</i></p> <p>Data indicating the plug-in's name and lifecycle callbacks for initialization and termination.</p> |
| Return Value | <p><i>true</i> if the handshake version number is valid, <i>false</i> if the handshake version number is unknown. If <i>false</i>, the plug-in is not initialized, and any capability it would add is not available.</p> |
| Header File | <i>PICommon.h</i> |
| Related Callbacks | PIExportHFTsProcType PIImportReplaceAndRegisterProcType PIInitProcType PIUnloadProcType |
| Related Methods | None |

PIImportReplaceAndRegisterProcType

```
ACCB1 ASBool ACCB2 PIImportReplaceAndRegisterProcType (void);
```

| | |
|--------------------------|---|
| Description | <p>(<i>Optional</i>) Callback for PIHandshake. This handshaking function is called by the viewer during initialization to allow a plug-in to import an HFT to another plug-in, replace an API method, or register for a notification.</p> <p>This is the <i>only</i> place that a plug-in may replace a function. You may call utility functions, but do not attempt to use Cos, PDMModel or AcroView-level methods here.</p> <p>If this callback returns <i>false</i>, the PIUnloadProcType callback is called. The plug-in <i>must</i> remove and release <i>all</i> menu items and other user interface elements, HFT's, HFTServers, release any memory or any other resources allocated, and so on, in its PIUnloadProcType callback.</p> |
| Parameters | None |
| Return Value | <i>true</i> if the HFT was imported successfully, <i>false</i> otherwise. |
| Header File | <i>PIVersn.h</i> |
| Related Callbacks | PIExportHFTsProcType PIHandshake |
| Related Methods | None |

PIInitProcType

```
ACCB1 ASBool ACCB2 PIInitProcType (void);
```

Description

(Required) Callback for [PIHandshake](#). This handshaking function is called by the viewer during initialization to allow a plug-in to do any sort of initialization it requires, such as adding user interface elements like menu items.

It is *not* safe to assume that all other plug-ins have initialized at this point.

If you want to do something after *all* plug-ins have been loaded and after the user has dismissed the open file dialog (if any), register for the notification *AVAppDidInitialize* and perform the action in the callback you register. Or register for an idle proc with *AVAppRegisterIdleProc* and perform the action in the callback you register. An example of such a case is adding a menu item to a menu or submenu added by another plug-in.

If this callback returns *false*, the [PIUnloadProcType](#) callback is called. The plug-in *must* remove and release *all* menu items and other user interface elements, HFT's, HFTServers, release any memory or any other resources allocated, and so on, in its [PIUnloadProcType](#) callback.

Parameters

None

Return Value

true if the plug-in initialized successfully, *false* otherwise.

Header File

PIVersn.h

Related Callbacks

[PIHandshake](#)

Related Methods

None

PIUnloadProcType

```
ACCB1 ASBool ACCB2 PIUnloadProcType (void);
```

| | |
|--------------------------|--|
| Description | <p>(<i>Optional</i>) Callback for PIHandshake. This handshaking function is called by the viewer when the plug-in unloads to allow it to perform any sort of cleanup needed. Use this routine to release any system resources you may have allocated.</p> <p>This is called for every plug-in when the viewer terminates or when any of the other PIHandshake callbacks return <i>false</i>.</p> <p>The plug-in <i>must</i> remove and release <i>all</i> menu items and other user interface elements, HFT's, HFTServers, release any memory or any other resources allocated, and so on, in its PIUnloadProcType callback.</p> |
| Parameters | None |
| Return Value | <i>true</i> if the plug-in finished its cleanup successfully, <i>false</i> otherwise. |
| Header File | <i>PIVersn.h</i> |
| Related Callbacks | PIHandshake |
| Related Methods | None |

PMBeginOperationProc

```
ACCB1 void ACCB2 PMBeginOperationProc (void* clientData);
```

| | |
|--------------------------|--|
| Description | Callback used in ProgressMonitor . Initialize the progress monitor and displays it with a current value of zero. This method must be called first when the progress monitor is used. |
| Parameters | <i>clientData</i> User-supplied data that was passed in the call to whatever API method required the progress monitor. |
| Return Value | None |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | PMEndOperationProc |

PMEndOperationProc

```
ACCB1 void ACCB2 PMEndOperationProc (void* clientData);
```

| | |
|--------------------------|--|
| Description | Callback used in ProgressMonitor . Draws the progress monitor with its current value set to the progress monitor's duration (a full progress monitor), then removes the progress monitor from the display. |
| Parameters | <i>clientData</i> User-supplied data that was passed in the call to whatever API method required the progress monitor. |
| Return Value | None |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | PMBeginOperationProc |

PMGetCurrValueProc

```
ACCB1 ASInt32 ACCB2 PMGetCurrValueProc (void* clientData);
```

| | |
|--------------------------|--|
| Description | Callback used in ProgressMonitor . Gets the progress monitor's duration, set by the most recent call the progress monitor's PMSetCurrValueProc . |
| Parameters | <i>clientData</i> User-supplied data that was passed in the call to whatever API method required the progress monitor. |
| Return Value | None |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | PMSetCurrValueProc |

PMGetDurationProc

```
ACCB1 ASInt32 ACCB2 PMGetDurationProc (void* clientData);
```

| | |
|--------------------------|---|
| Description | Callback used in ProgressMonitor . Gets the progress monitor's duration, set by the most recent call the progress monitor's PMSetDurationProc . |
| Parameters | <i>clientData</i> User-supplied data that was passed in the call to whatever API method required the progress monitor. |
| Return Value | The progress monitor's maximum value. |
| Header File | <i>ASExpT.h</i> |
| Related Callbacks | PMSetDurationProc |

PMSetCurrValueProc

```
ACCB1 void ACCB2 PMSetCurrValueProc (ASInt32 currValue,  
    void* clientData);
```

Description Callback used in [ProgressMonitor](#). Sets the current value of the progress monitor and updates the display. The allowed value ranges from 0 (empty) to the value passed to *setDuration*. For example, if the progress monitor's duration is 10, the current value must be between 0 and 10, inclusive.

Parameters *currValue*
 The progress monitor's current value.

clientData
 User-supplied data that was passed in the call to whatever API method required the progress monitor.

Return Value None

Header File *ASExpT.h*

Related Callbacks [PMGetCurrValueProc](#)

PMSetDurationProc

```
ACCB1 void ACCB2 PMSetDurationProc (ASInt32 duration,  
    void* clientData);
```

Description Callback used in [ProgressMonitor](#). Sets the value that corresponds to a full progress monitor display. The progress monitor is subsequently filled in by setting its current value. This method must be called before you can set the progress monitor's current value.

Parameters *duration*
The maximum value the progress monitor will be allowed to have.

clientData
User-supplied data that was passed in the call to whatever API method required the progress monitor.

Return Value None

Header File *ASExpT.h*

Related Callbacks [PMGetDurationProc](#)

TKResourceAcquireProc

```
ACCB1 void* ACCB2* TKResourceAcquireProc (char* resourceName,
    ASInt32 resType, void* registry, ASInt32* size,
    void* clientData, ASStm* rdStm);
```

Description Acquires the specified resource and use it to fill in the *rdStm* parameter.

Parameters

resourceName

Name of resource to acquire.

resType

One of the following from the **resType** enum:
resCMapData, **resCharTypeData**,
resCharRangeData, **resFontPropData**,
resCJKNoteFontNormal, **resCJKNoteFontBold**,
resCJKNoteFontOblique, **resFontData**, or
resCMapLastID.

registry

Must be one of the following.

"Adobe-Japan1" — Japanese

"Adobe-Korea1" — Korean

"Adobe-CNS1" — Traditional Chinese

"Adobe-GB1" — Simplified Chinese

size

(Filled by the method) Size of the *ASStm* returned.

clientData

The *clientData* specified in the [TKResourceProcs](#) structure.

rdStm

(Filled by the method) An *ASStm* for the data requested.

Return Value

A *void** of data that you want returned to you in [TKResourceReleaseProc](#). This data can be used to determine what Resource to release. You can return *NULL* from [TKResourceAcquireProc](#).

Header File

PDFInit.h

Related Callbacks

[TKResourceReleaseProc](#)

Related Methods

[PDFLInit](#)

TKResourceReleaseProc

```
ACCB1 void ACCB2* TKResourceReleaseProc (ASStm rdStm,  
void* data, void* clientData);
```

| | |
|--------------------------|--|
| Description | Releases the resources previously acquired and closes the <i>ASStm</i> passed in as <i>rdStm</i> . |
| Parameters | <p><i>rdStm</i></p> <p>An <i>ASStm</i> for the resource to release and close with ASStmClose.</p> <p><i>data</i></p> <p>The <i>clientData</i> returned from TKResourceAcquireProc.</p> <p><i>clientData</i></p> <p>The <i>clientData</i> specified in the TKResourceProcs structure.</p> |
| Return Value | None |
| Header File | <i>PDFInit.h</i> |
| Related Callbacks | TKResourceAcquireProc |
| Related Methods | PDFLInit |

Declarations

AGMALABColorRec

```
typedef struct _t_AGMALABColorRec {  
    uint8 alpha;  
    uint8 l;  
    uint8 a;  
    uint8 b;  
} AGMLABColorRec;
```

Description Data structure representing a Lab color value.

Header File *AGMImage.h*

Related Callbacks None

Related Methods None

alpha

Alpha channel intensity, which indicates degree of transparency. Must be a number between 0 (minimum intensity) and 255 (maximum intensity).

l, a, b

L*, a* and b* components of the color. Each must be a number between 0 and 255.

AGMARGBColorRec

```
typedef struct _t_AGMARGBColorRec {  
    uint8 alpha;  
    uint8 red;  
    uint8 green;  
    uint8 blue;  
} AGMARGBColorRec;
```

Description Data structure representing an RGB color value.

Header File *AGMImage.h*

Related Callbacks None

Related Methods None

alpha

Alpha channel intensity, which indicates degree of transparency. Must be a number between 0 (minimum intensity) and 255 (maximum intensity).

red, green, blue

Color component intensity. Must be a number between 0 (minimum intensity) and 255 (maximum intensity).

AGMBlackPointFlt

```
typedef AGMXYZColorFlt AGMBlackPointFlt;
```

| | |
|--------------------------|---|
| Description | An AGMXYZColorFlt value that specifies the CIE 1931 (XYZ)-space tristimulus value of the diffuse black point. The numbers must be non-negative. The default value is [0 0 0]. See Section 4.8.3 in the <i>PostScript Language Reference Manual, Second Edition</i> for further details. |
| Header File | <i>PEExpt.h</i> |
| Related Types | AGMWhitePointFlt |
| Related Callbacks | None |
| Related Methods | None |

AGMCMYKColorRec

```
typedef struct _t_AGMCMYKColorRec {  
    uint8 cyan;  
    uint8 magenta;  
    uint8 yellow;  
    uint8 black;  
} AGMCMYKColorRec;
```

Description Data structure representing a CMYK color value.

Header File *AGMImage.h*

Related Callbacks None

Related Methods None

cyan, magenta, yellow, black

Color component amount. Must be a number between 0 (no ink) and 255 (maximum ink).

AGMColorRangeFlt

```
typedef struct {  
    float min;  
    float max;  
} AGMColorRangeFlt;
```

| | |
|--------------------------|--|
| Description | A range of values for a* and b* components in an AGMLabCalFlt color space. |
| Header File | <i>PEExpt.h</i> |
| Related Callbacks | None |
| Related Methods | None |

| | |
|------------|--|
| <i>min</i> | Minimum floating point value in color range. |
| <i>max</i> | Maximum floating point value in color range. |

AGMColorTab

```
typedef struct _t_AGMColorTab {  
    int32 numColors;  
    void* theColors;  
} AGMColorTab;
```

Description Data structure representing a color table.

Header File *AGMImage.h*

Related Callbacks None

Related Methods None

numColors

Number of colors in the table.

theColors

Pointer to the structure representing the color table, which is one of:

[AGMALABColorRec](#)

[AGMRGBColorRec](#)

[AGMCMYKColorRec](#)

[AGMLABColorRec](#)

[AGMRGBColorRec](#)

AGMFixedMatrix

```
typedef struct _t_AGMFixedMatrix {  
    ASFixed a;  
    ASFixed b;  
    ASFixed c;  
    ASFixed d;  
    ASFixed e;  
    ASFixed f;  
} AGMFixedMatrix;
```

| | |
|--------------------------|---|
| Description | Data structure representing a matrix of fixed-point numbers. This matrix is generally used as a transformation matrix. See Section 3.10 in the Portable Document Format Reference Manual for a discussion of transformation matrices. |
| Header File | <i>AGMTypes.h</i> |
| Related Callbacks | None |
| Related Methods | AGMConcat |

a, b, c, d, e, f

The matrix's elements.

AGMFixedPoint

```
typedef struct _t_AGMFixedPoint {  
    ASFixed x;  
    ASFixed y;  
} AGMFixedPoint;
```

Description Data structure representing a point.

Header File *AGMTypes.h*

Related Callbacks None

Related Methods [AGMLineTo](#)
[AGMMoveTo](#)

| | |
|----------|---------------------------|
| x | The point's x-coordinate. |
|----------|---------------------------|

| | |
|----------|---------------------------|
| y | The point's y-coordinate. |
|----------|---------------------------|

AGMGrayCalFlt

```
typedef struct _t_AGMGrayCalFlt {  
    AGMWhitePointFlt whitePoint;  
    AGMBlackPointFlt blackPoint;  
    float gamma;  
} AGMGrayCalFlt;
```

| | |
|--------------------------|---|
| Description | A CalGray color space, as defined in Section 7.11.4, "CalGray color spaces" in the Portable Document Format Reference Manual . |
| Header File | <i>PEExpt.h</i> |
| Related Types | PDEGrayCalFlt |
| Related Callbacks | None |
| Related Methods | None |

| |
|--|
| <i>whitePoint</i> |
| The CIE 1931 (XYZ)-space tristimulus value of the diffuse white point. |
| <i>blackPoint</i> |
| The CIE 1931 (XYZ)-space tristimulus value of the diffuse black point. |
| <i>gamma</i> |
| Gamma value. Defines the exponential relationship between the gray component and Y. <i>gamma</i> must be positive and is generally greater than or equal to 1. The default value is 1. |

AGMImageAlphaRecord

```
struct _t_AGMImageAlphaRecord {  
    AGMInt16Rect bounds;  
    void* baseAddr;  
    int32 byteWidth;  
    int16 colorSpace;  
    int16 bitsPerPixel;  
    ASFixed* decodeArray;  
    AGMColorTab colorTab;  
    void* alphaBaseAddr;  
    int32 alphaByteWidth;  
    int32 alphaBitsPerPixel;  
    ASFixed* alphaDecode;  
};
```

Description

Data structure representing an image. This record supports images that have no alpha channel, that have alpha data interleaved with pixel data, or that have a separate alpha channel.

Note: The Placed PDF Library only supports a record that has no alpha channel. When initializing the record, the alphaBaseAddr, alphaByteWidth, alphaBitsPerPixel, and alphaDecode values must all be zero.

Header File

AGMImage.h

Related Callbacks

None

Related Methods

[AGMNewRasterDev](#)

bounds

Specifies the physical size and logical coordinates of the image data. The *xMin*, *yMin*, *xMax*, and *yMax* fields indicate the logical origin and bounds of the image. The size of the image rectangle is *xMax* - *xMin* in width by *yMax* - *yMin* in height.

baseAddr

Pointer to the image data. Samples are fully packed within a scan line. When used with an *AGMRasterDevice*, the rows must be 32-bit aligned.

byteWidth

Specifies the number of bytes of pixel data in each row. When used with an *AGMRasterDevice*, must be a multiple of 4 to indicate that the data is 32-bit aligned.

colorSpace

Specifies the color space and order of the data. Must be one of *kGrayColorSpace*, *kRGBColorSpace*, *kCMYKColorSpace*, or *kLabColorSpace*.

bitsPerPixel

Specifies the number of bits used to represent each pixel in the image. This value is calculated by multiplying the bits per component, which may be 1, 2, 4, or 8, by the number of components. For *kGrayColorSpace* the number of components is 1, for *kRGBColorSpace* or *kLabColorSpace* the number of components is 3, and for *kCMYKColorSpace* the number of components is 4.

decodeArray

Currently unused. Set to zero.

colorTab

An [AGMColorTab](#) specifying a color table for an image that uses an indexed color space. In an indexed color space, there is only one component, which is the index into the color table.

alphaBaseAddr

Currently unused. Set to zero.

alphaByteWidth

Currently unused. Set to zero.

alphaBitsPerPixel

Currently unused. Set to zero.

alphaDecode

Currently unused. Set to zero.

AGMInt16Rect

```
typedef struct _t_AGMInt16Rect {  
    int16 xMin;  
    int16 yMin;  
    int16 xMax;  
    int16 yMax;  
} AGMInt16Rect;
```

Description Data structure representing a rectangle.

Header File *AGMTypes.h*

Related Callbacks None

Related Methods None

xMin

The rectangle's minimum *x*-coordinate.

yMin

The rectangle's minimum *y*-coordinate.

xMax

The rectangle's maximum *x*-coordinate.

yMax

The rectangle's maximum *y*-coordinate.

AGMLabCalFlt

```
typedef struct _t_AGMLabCalFlt {  
    AGMWhitePointFlt whitePoint;  
    AGMBlackPointFlt blackPoint;  
    AGMColorRangeFlt rangeA, rangeB;  
} AGMLabCalFlt;
```

Description A **L*a*b*** color space, as defined in Section 7.11.6, “Lab color spaces” in the [Portable Document Format Reference Manual](#).

Header File *PEExpt.h*

Related Types [PDELabCalFlt](#)

Related Callbacks None

Related Methods None

whitePoint

The CIE 1931 (XYZ)-space tristimulus value of the diffuse white point.

blackPoint

The CIE 1931 (XYZ)-space tristimulus value of the diffuse black point.

rangeA

The range of the a* component. The default values are {-100, 100}.

rangeB

The range of the b* component. The default values are {-100, 100}.

AGMLABColorRec

```
typedef struct _t_AGMLABColorRec {  
    uint8 mustBeZero;  
    uint8 l;  
    uint8 a;  
    uint8 b;  
} AGMLABColorRec;
```

Description Data structure representing a Lab color value.

Header File *AGMImage.h*

Related Callbacks None

Related Methods None

mustBeZero

Must be 0.

l, a, b

L*, a* and b* components of the color. Must be a number between 0 and 255.

AGMMemObj

```
typedef struct _t_AGMMemObj {
    AGMMemAllocator alloc;
    AGMMemDeleter free;
    void* userData;
} AGMMemObj;
```

Description Data structure containing callbacks to allocate and free memory.

By having client applications provide their own memory allocation/free callbacks, the application can better manage its own storage space. Not all of the memory used by the Placed PDF Library is allocated and freed using these callbacks, so the client application is not in complete control of its memory.

Header File *AGMemObj.h*

Related Callbacks [AGMMemAllocator](#)
[AGMMemDeleter](#)

Related Methods [AGMNewBitmapPort](#)
[AGMNewRasterDev](#)
[AGMNewRasterPort](#)
[AGMNewWindowPort](#)

AGMMemAllocator

Callback for allocating memory.

AGMMemDeleter

Callback for freeing memory.

userData

User-supplied data to pass to [AGMMemAllocator](#) and [AGMMemDeleter](#) each time they are called.

AGMRGBCalFlt

```
typedef struct _t_AGMRGBCalFlt {  
    AGMWhitePointFlt whitePoint;  
    AGMBlackPointFlt blackPoint;  
    float redGamma;  
    float greenGamma;  
    float blueGamma;  
    float matrix[9];  
} AGMRGBCalFlt;
```

| | |
|--------------------------|---|
| Description | A CalRGB color space, as defined in Section 7.11.5, "CalRGB color spaces" in the Portable Document Format Reference Manual . |
| Header File | <i>PEExpt.h</i> |
| Related Types | PDERGBCalFlt |
| Related Callbacks | None |
| Related Methods | None |

| |
|--|
| <i>whitePoint</i> |
| The CIE 1931 (XYZ)-space tristimulus value of the diffuse white point. |
| <i>blackPoint</i> |
| The CIE 1931 (XYZ)-space tristimulus value of the diffuse black point. |
| <i>redGamma, greenGamma, blueGamma</i> |
| Input calibrated RGB values that specify the gamma for the red, green, and blue components respectively. The default values are {1, 1, 1}. |
| <i>matrix</i> |
| Nine numbers that specify the linear interpretation of the gamma-modified red, green, and blue components. The default values are {1, 0, 0, 0, 1, 0, 0, 0, 1}. |

AGMRGBColorRec

```
typedef struct _t_AGMRGBColorRec {  
    uint8 mustBeZero;  
    uint8 red;  
    uint8 green;  
    uint8 blue;  
} AGMRGBColorRec;
```

Description Data structure representing an RGB color value.

Header File *AGMImage.h*

Related Callbacks None

Related Methods None

mustBeZero

Must be 0.

red, green, blue

Color component intensity. Must be a number between 0 (minimum intensity) and 255 (maximum intensity).

AGMWhitePointFlt

```
typedef AGMXYZColorFlt AGMWhitePointFlt;
```

| | |
|--------------------------|---|
| Description | An AGMXYZColorFlt value that specifies the CIE 1931 (XYZ)-space tristimulus value of the diffuse white point. The numbers must be non-negative. The default value is [0 0 0]. See the discussion in Section 4.8.3 in the <i>PostScript Language Reference Manual, Second Edition</i> for further details. |
| Header File | <i>PEExpt.h</i> |
| Related Types | AGMBlackPointFlt |
| Related Callbacks | None |
| Related Methods | None |

AGMXYZColorFlt

```
typedef struct {  
    float x;  
    float y;  
    float z;  
} AGMXYZColorFlt;
```

| | |
|--------------------------|---|
| Description | A point in a 3-dimensional color space. |
| Header File | <i>PEExpt.h</i> |
| Related Callbacks | None |
| Related Methods | None |

x, y, z

Floating point numbers representing components in a 3-dimensional color space.

ASFile Flags

| | |
|------------------------|---|
| Description | Flags to describe file transfers. Set by external file systems. |
| Header File | <i>ASExpT.h</i> |
| Related Methods | ASFileSysOpenFile |

| Mode | Description |
|----------------------------|---|
| <i>kASFileSlowTransfer</i> | The file's data transfer rate is generally slow, for example, because it is on a floppy disk or being accessed via modem. |
| <i>kASFileSlowConnect</i> | Initiating each access to the file is slow, for example, because the file is being served by an HTTP server that spawns a new process for each request. |
| <i>kASFileUseMRead</i> | Use multiread commands to access the file. |

ASFile Open Modes

| | |
|------------------------|---|
| Description | File access modes used to specify how a file can be used when it is open. Not all modes can be specified individually; <i>ASFILE_CREATE</i> can be used only in conjunction with <i>ASFILE_READ</i> or <i>ASFILE_WRITE</i> . In addition, it is acceptable to specify <i>ASFILE_READ</i> and <i>ASFILE_WRITE</i> together, by OR'ing the two constants. <i>ASFILE_SERIAL</i> and <i>ASFILE_LOCAL</i> (present only in version 3.0 or later) are hints that help the Acrobat viewer optimize access to the file; they must be OR'ed with one or more of the other constants. |
| Header File | <i>ASExpT.h</i> |
| Related Methods | ASFileSysOpenFile ASFileReopen |

| Mode | Description |
|----------------------|---|
| <i>ASFILE_READ</i> | Open the file for reading. |
| <i>ASFILE_WRITE</i> | Open the file for writing. |
| <i>ASFILE_CREATE</i> | Create the file if it does not exist. |
| <i>ASFILE_SERIAL</i> | A hint indicating that the file will be accessed sequentially. |
| <i>ASFILE_LOCAL</i> | A hint indicating that a local copy of the file will be needed. |

ASFileMode Flags

Description Flags to describe file modes.

Header File *ASExpT.h*

Related Methods [ASFileRead](#)
[ASFileSetMode](#)

| Mode | Description |
|---|---|
| <i>kASFileModeDoNotYieldIfBytesNotReady</i> | If set, then ASFileRead does <i>not</i> yield if bytes are not ready (which raises the fileErrBytesNotReady exception). |

ASFileStatus Flags

| | |
|------------------------|---|
| Description | Values returned by ASFileSysGetStatusProc . |
| Header File | <i>ASExpT.h</i> |
| Related Methods | ASFileRead |

| Mode | Description |
|-----------------------------|--|
| <i>kASFileOkay</i> | The <i>MDFile</i> is in a valid state. |
| <i>kASFileIsTerminating</i> | The <i>MDFile</i> is being closed, for example, because the file is being displayed in a Web browser's window and the user canceled downloading. |

ASFileSysRec

```
typedef struct _t_ASFileSysRec {
    ASSize_t size;
    ASFileSysOpenProc open;
    ASFileSysCloseProc close;
    ASFileSysFlushProc flush;
    ASFileSysSetPosProc setpos;
    ASFileSysGetPosProc getpos;
    ASFileSysSetEofProc seteof;
    ASFileSysGetEofProc geteof;
    ASFileSysReadProc read;
    ASFileSysWriteProc write;
    ASFileSysRemoveProc remove;
    ASFileSysRenameProc rename;
    ASFileSysIsSameFileProc isSameFile;
    ASFileSysGetNameProc getName;
    ASFileSysGetTempPathNameProc getTempPathName;
    ASFileSysCopyPathNameProc copyPathName;
    ASFileSysDiPathFromPathProc diPathFromPath;
    ASFileSysPathFromDIPathProc pathFromDIPath;
    ASFileSysDisposePathNameProc disposePathName;
    ASFileSysGetFileSysNameProc getFileSysName;
    ASFileSysGetStorageFreeSpaceProc getStorageFreeSpace;
    ASFileSysFlushVolumeProc flushVolume;

    /* The following are present only in version 3.0
    or later */
    ASFileSysGetFileFlags getFileFlags;
    ASFileSysAsyncReadProc readAsync;
    ASFileSysAsyncWriteProc writeAsync;
    ASFileSysAsyncAbortProc abortAsync;
    ASFileSysYieldProc yield;
    ASFileSysMReadRequestProc mreadRequest;
    ASFileSysGetStatusProc getStatus;
    ASFileSysCreatePathNameProc createPathName;
    ASFileSysAcquireFileSysPathProc acquireFileSysPath;
    ASFileSysClearOutstandingMReadsProc clearOutstandingMReads;
} ASFileSysRec, *ASFileSys;
```

| | |
|------------------------|---|
| Description | Data structure containing callbacks that implement a file system. |
| Header File | <i>ASExpT.h</i> |
| Related Methods | Numerous. See ASFileSys . |

size

Size of the data structure. Must be set to *sizeof(ASFileSysRec)*.

ASFixed

ASFixedP

```
typedef ASInt32 ASFixed, *ASFixedP;
```

Description

The *ASFixed* type is a 32-bit quantity representing a rational number with the high (low on little-endian machines) 16 bits representing the number's mantissa and the low (high) 16 bits representing the fractional part. The definition is platform-dependent.

ASFixedP is a pointer to an *ASFixed*.

Addition, subtraction, and negation with *ASFixed* types can be done with + and -, unless you care about overflow, in which case you should use *FixedSum* and *FixedDiff*.

Overflow in *ASFixed*-value operations is indicated by the values *fixedPositiveInfinity* and *fixedNegativeInfinity*.

Header Files

Platform-dependent

Related Methods

Numerous

ASFixedMatrix

ASFixedMatrixP

```
typedef struct _t_FixedMatrix {  
    ASFixed a;  
    ASFixed b;  
    ASFixed c;  
    ASFixed d;  
    ASFixed h;  
    ASFixed v;  
} ASFixedMatrix, *ASFixedMatrixP;
```

| | |
|------------------------|----------------------------------|
| Description | Matrix containing fixed numbers. |
| Header File | <i>ASExpT.h</i> |
| Related Methods | Numerous |

ASFixedPoint

ASFixedPointP

```
typedef struct _t_FixedPoint {  
    ASFixed h;  
    ASFixed v;  
} ASFixedPoint, *ASFixedPointP;
```

| | |
|------------------------|--|
| Description | Point (in two-dimensional space) represented by two fixed numbers. |
| Header File | <i>ASExpT.h</i> |
| Related Methods | Numerous |

ASFixedQuad

ASFixedQuadP

```
typedef struct _t_ASFixedQuad {  
    ASFixedPoint tl, tr, bl, br;  
} ASFixedQuad, *ASFixedQuadP;
```

| | |
|------------------------|--|
| Description | Quadrilateral represented by four fixed points (one at each corner). In the Acrobat viewer, a quadrilateral differs from a rectangle in that the latter must always have horizontal and vertical sides, and opposite sides must be parallel. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | Numerous |

ASFixedRect

ASFixedRectP

```
typedef struct _t_ASFixedRect {  
    ASFixed left;  
    ASFixed top;  
    ASFixed right;  
    ASFixed bottom;  
} ASFixedRect, *ASFixedRectP;
```

| | |
|------------------------|---|
| Description | A rectangle represented by the coordinates of its four sides. In the Acrobat viewer, a rectangle differs from a quadrilateral in that the former must always have horizontal and vertical sides, and opposite sides must be parallel. |
| Header File | <i>ASExpT.h</i> |
| Related Methods | Numerous |

ASIORequest

```
typedef struct _t_ASIORequestRec {
    MDFile mdFile;
    void* ptr;
    ASInt32 offset;
    ASInt32 count;
    ASInt32 totalBytesCompleted;
    ASInt32 pError;
    void* clientData;
    ASIODoneProc IODoneProc;
    void* IODoneProcData;
} ASIORequestRec, *ASIORequest;
```

Description Data structure representing an I/O request.

Header File *ASExpT.h*

Related Callbacks [ASFileSysAsyncReadProc](#)
[ASFileSysAsyncWriteProc](#)
[ASIODoneProc](#)

| | |
|----------------------------|---|
| <i>mdFile</i> | The <i>MDFile</i> corresponding to the <i>ASFile</i> this request is for. |
| <i>ptr</i> | Pointer to data to write to or read from <i>mdFile</i> . |
| <i>offset</i> | Offset (specified in bytes) into <i>mdFile</i> of the first byte to read or write. |
| <i>count</i> | Number of bytes to read/write. Must be filled in before <i>IODoneProc</i> is called. If zero, the read was either terminated or did not complete. |
| <i>totalBytesCompleted</i> | Number of bytes actually read or written. |
| <i>pError</i> | Error code. This code is filled by the <i>ASFileSys</i> before <i>IODoneProc</i> is called. If nonzero, the read was either terminated or did not complete. |
| <i>clientData</i> | User-supplied data that the <i>ASFileSys</i> can use for anything it wishes. |

IODoneProc

User-supplied callback to call by the *ASFileSys* when the operation has completed. If non-*NULL*, it points to a procedure that *must* be called either when the request has terminated due to error or other condition, or when all of the bytes have been received for this request.

Note: This callback may be called at interrupt time.

IODoneProcData

User-supplied that is available for *IODoneProc* to use.

ASPlatformPrinterSpec

```

/* In Mac OS */

typedef struct _t_ASPlatformPrinterSpec *ASPlatformPrinterSpec;

typedef struct _t_ASPlatformPrinterSpec {
    ASSize_t size;
    void* cGrafPtr;
    short hRes, vRes;
} ASPlatformPrinterSpecRec;

/* In UNIX */

typedef struct _t_ASPlatformPrinterSpec *ASPlatformPrinterSpec;

typedef struct _t_ASPlatformPrinterSpec {
    ASSize_t size;
    char* printerName;
    ASUns8* baseAddr;
    ASUns32 rowBytes;
    ASUns32 depth;
    AVRect32 bounds;
} ASPlatformPrinterSpecRec;

/* In Windows */

#define kPrinterSpecNameLen 64 /* room for 32 Unicode chars */

typedef struct _t_ASPlatformPrinterSpec *ASPlatformPrinterSpec;

typedef struct _t_ASPlatformPrinterSpec {
    ASSize_t size;
    char driverName[kPrinterSpecNameLen];
    char printerName[kPrinterSpecNameLen];
    char portName[kPrinterSpecNameLen];
    ASBool createMetaFile;
    char metaFileName[260];
    ASInt32 win16Hdc;
    ASInt32 hRes;
    ASInt32 vRes;
    ASInt32 colorDepth;
    ASBool isPostScript;
} ASPlatformPrinterSpecRec;

```

| | |
|------------------------|--|
| Description | Data structure representing a platform specification for a printer. Used in AVDocPrintParams . |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocPrintPagesWithParams |

| For all platforms |
|--|
| <p><i>size</i></p> <p>Size of the data structure. Must be set to <code>sizeof(ASPlatformPrinterSpecRec)</code>.</p> |
| In Mac OS |
| <p><i>cGrafPtr</i></p> <p>Port to print to.</p> |
| <p><i>hRes, vRes</i></p> <p>Best known resolution of current printer.</p> |
| In UNIX |
| <p><i>printerName</i></p> <p>Print command, such as "lp -dMyPrinter -n4". If <i>printerName</i> is <i>NULL</i>, a default print command is used. The Acrobat viewer's built-in default is "lp" on most UNIX systems, and "lpr" on SunOS. This should print to the system's default printer. Some UNIX systems also look at the environment variable <i>LPDEST</i> or <i>PRINTER</i>. See the documentation for your platform to determine whether or not this is the case.</p> |
| <p><i>baseAddr</i></p> <p>Currently unused.</p> |
| <p><i>rowBytes</i></p> <p>Currently unused.</p> |
| <p><i>depth</i></p> <p>Currently unused.</p> |
| <p><i>bounds</i></p> <p>Currently unused.</p> |
| In Windows |
| <p>The following items should be provided for full, non-interactive printing.</p> |
| <p><i>driverName</i></p> <p>See <i>Windows.h DEVNAMES</i> for a description of these fields.</p> |

| In Windows | |
|--|---|
| <i>printerName</i> | Name of the printer. For example, "HPPCL," "HP LaserJet 4," or "LPT1." |
| <i>portName</i> | Port to print to. |
| For embedded printing, <i>createMetaFile</i> , <i>metaFileName</i> , <i>win16Hdc</i> , <i>hRes</i> , <i>vRes</i> , <i>colorDepth</i> , and <i>isPostScript</i> must be provided. | |
| <i>createMetaFile</i> | Must be <i>true</i> if Windows 32-bit platforms; optional for Windows 16-bit platforms. |
| <i>metaFileName</i> | Pathname for the metafile. Only required if <i>createMetaFile</i> is <i>true</i> . |
| <i>win16Hdc</i> | Device context for 16- or 32-bit platforms. |
| <i>hRes</i> | Horizontal resolution of printer; 300 dpi, for example. |
| <i>vRes</i> | Vertical resolution of printer; 300 dpi, for example. |
| <i>colorDepth</i> | Color depth of device; typically 1, 8, or 24. This determines the depth of images created for the printer. You may specify 24 when printing to a monochrome printer. The driver is expected to convert to the printer depth. Not used if <i>isPostScript</i> is <i>true</i> . |
| <i>isPostScript</i> | Set to <i>true</i> if printing to a PostScript printer. |

ASTimeRec

ASTimeRecP

```
typedef struct _t_ASTimeRec {
    ASInt16 year;
    ASInt16 month;
    ASInt16 date;
    ASInt16 hour;
    ASInt16 minute;
    ASInt16 second;
    ASInt16 millisecond;
    ASInt16 day;
    ASInt16 gmtOffset;
} ASTimeRec, *ASTimeRecP;
```

| | |
|------------------------|---|
| Description | Time/Date structure. The <i>millisecond</i> field is currently unused. |
| Header File | <i>ASExpT.h</i> |
| Related Methods | PDAnnotGetDate PDAnnotSetDate |

AS Types

```
typedef boolean ASBool;  
typedef Uns8 ASUns8;  
typedef Uns16 ASUns16;  
typedef Uns32 ASUns32;  
typedef Int8 ASInt8;  
typedef Int16 ASInt16;  
typedef Int32 ASInt32;
```

| | |
|------------------------|---|
| Description | Types provided for platform independence. |
| Header File | <i>CoreExpt.h</i> |
| Related Methods | Numerous |

AVActionHandlerProcs

```
typedef struct _t_AVActionHandlerProcs {
    ASSize_t size;
    AVActionPerformProc Perform;
    AVActionDoPropertiesProc DoProperties;
    AVActionFillActionDictProc FillActionDict;
    AVActionGetInstructionsProc GetInstructions;
    AVActionGetButtonTextProc GetButtonText;
    AVActionGetStringOneTextProc GetStringOneText;
    AVActionGetStringTwoTextProc GetStringTwoText;
    AVActionCopyProc Copy; /* New for Acrobat 3.01 */
} AVActionHandlerProcsRec, *AVActionHandlerProcs;
```

Description Data structure containing callbacks that implement an action handler. The callbacks implement the action handler functions. For example, display user interface text, request the action's properties from the user, perform the action.

Header File *AVExpT.h*




Related Methods [AVAppRegisterActionHandler](#)
[AVActionHandlerGetProcs](#)
[AVDocCopyAnnot](#)

size

Size of the data structure. Must be set to `sizeof(AVActionHandlerProcsRec)`.

AVAlert Icons

| | |
|------------------------|---|
| Description | Standard icons used in alert boxes. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | <u>AVAlert</u> <u>AVAlertConfirm</u> <u>AVAlertNote</u> |

| iconType | Icon (Macintosh) |
|----------------------|---|
| <i>ALERT_NOICON</i> | No icon. |
| <i>ALERT_STOP</i> |  |
| <i>ALERT_CAUTION</i> |  |
| <i>ALERT_NOTE</i> |  |

AVAnnotHandler

```
typedef struct _t_AVAnnotHandler {
    ASSize_t size;
    ASUns32 flags;
    AVAnnotHandlerDoClickProc DoClick;
    AVAnnotHandlerAdjustCursorProc AdjustCursor;
    AVAnnotHandlerPtInAnnotViewBBoxProc PtInAnnotViewBBox;
    AVAnnotHandlerGetAnnotViewBBoxProc GetAnnotViewBBox;
    AVAnnotHandlerNotifyAnnotRemovedFromSelectionProc
    NotifyAnnotRemovedFromSelection;
    AVAnnotHandlerNotifyAnnotAddedToSelectionProc
    NotifyAnnotAddedToSelection;
    AVAnnotHandlerDrawProc Draw;
    AVAnnotHandlerNewProc New;
    AVAnnotHandlerGetTypeProc GetType;
    AVAnnotHandlerNotifyDestroyProc NotifyDestroy;
    AVAnnotHandlerDoPropertiesProc DoProperties;
    AVAnnotHandlerDoKeyDownProc DoKeyDown;
    AVAnnotHandlerGetLayerProc GetLayer;
    /* New callbacks in Acrobat 3.0 */
    AVAnnotHandlerCursorEnterProc CursorEnter;
    AVAnnotHandlerCursorExitProc CursorExit;
    /* New callbacks in Acrobat 3.01 */
    AVAnnotHandlerCopyProc Copy;
    /* New callbacks in Acrobat 4.0 */
    AVAnnotHandlerDoClickProc DoRightClick;
    AVAnnotHandlerGetInfoProc GetInfo;
    AVAnnotHandlerDeleteInfoProc DeleteInfo;
} AVAnnotHandlerRec, *AVAnnotHandler;
```

Description Data structure containing callbacks that implement an annotation handler. The callbacks implement the annotation handler functions. For example, draw the annotation, highlight the annotation when it is selected, and the data specifies properties of the annotation (for example, text selection behavior).

Header File *AVEXPT.h*

Related Callbacks [PDAnnotHandlerDeleteAnnotInfoProc](#)
[PDAnnotHandlerGetAnnotInfoFlagsProc](#)
[PDAnnotHandlerGetAnnotInfoProc](#)
[PDAnnotHandlerGetTypeProc](#)

Related Methods

[AVAppRegisterAnnotHandler](#)
[AVAppGetAnnotHandlerByName](#)
[AVDocCopyAnnot](#)

size

Size of the data structure. Must be set to `sizeof(AVAnnotHandlerRec)`.

flags

A collection of flags that affect the annotation's behavior. The flags may be ORed together.

Note: These flags are not the ones used in [PDAnnot Flags](#).

Permissible flags include:

ANNOT_CLIP_TEXT_SELECTION

If this flag is set, text selection in the main document never selects text within the annotation (that is, the annotation behaves like the Acrobat viewer's text annotation). If this flag is not set, text selection in the main document can select text within the annotation (that is, the annotation behaves like the Acrobat viewer's link annotation).

ANNOT_WANTS_SHIFT_KEY

This flag is set to prevent the standard shift-key ignores annotation's behavior.

AVAnnotHandlerInfo

```
typedef struct _t_AVAnnotHandlerInfoRec {
    ASSize_t size;
    unsigned char* cName;
    void* vBitmap;
} AVAnnotHandlerInfoRec, *AVAnnotHandlerInfo;
```

Description Structure used to describe information for a particular annotation type.

Header File *AVExpt.h*

Related Callbacks [AVAnnotHandlerDeleteInfoProc](#)
[AVAnnotHandlerGetInfoProc](#)

Related Methods [AVAnnotHandlerGetInfo](#)

| |
|---|
| <i>size</i> |
| Size of the data structure. Must be set to <i>sizeof(AVAnnotHandlerInfoRec)</i> . |
| <i>cName</i> |
| User interface name of annotation type in the host encoding. |
| <i>vBitmap</i> |
| Platform-dependent bitmap used as the annotation icon. If <i>NULL</i> , the annotation manager uses the unknown annotation icon for the annotation. |

AVAuxDataHandler

```
typedef struct _t_AVAuxDataHandler {  
    ASSize_t size;  
    AVAuxDataPerformProc PerformProc;  
} AVAuxDataHandlerRec, *AVAuxDataHandler;
```

Description Data structure containing callbacks and data representing an auxiliary data handler.

Header File *AVExpT.h*

Related Methods [AVDocSendAuxData](#)
[AVHasAuxDataHandler](#)
[AVRegisterAuxDataHandler](#)

| |
|---|
| <i>size</i> |
| Size of the data structure. Must be set to <i>sizeof(AVAuxDataHandlerRec)</i> . |
| <i>PerformProc</i> |
| Called with auxiliary data when a client calls AVDocSendAuxData . This proc should perform whatever action it needs to do for the auxiliary data. |

AVCursor

```
typedef struct _t_AVCursor *AVCursor;
```

| | |
|------------------------|---|
| Description | Data structure representing the cursor. See Predefined Cursors for a list of already defined cursor shapes. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVSysGetCursor AVSysGetStandardCursor AVSysSetCursor |

AVDestInfo

```
typedef struct _t_AVDestInfo {
    ASSize_t size;
    const char* namedDest;
    ASInt32 nameLength;
    ASInt32 pageNum;
    ASAtom fitType;
    ASFixedRect destRect;
    ASFixed zoom;
} AVDestInfoRec, *AVDestInfo;
```

Description

Data structure representing a destination in a PDF document. An *AVDestInfo* carries all the information that a [PDViewDestination](#) can. Used for ensuring that cross-document links in external windows act as expected, so a client can go to a destination without building it via [PDViewDestCreate](#), which doesn't work on read-only documents.

Header File

AVExpT.h

Related Methods

[AVPageViewToDestInfo](#)
[AVPageViewUseDestInfo](#)
[AVDestInfoDestroy](#)

| | |
|-------------------|--|
| <i>size</i> | Size of the data structure. Must be set to <i>sizeof(AVDestInfo)</i> . |
| <i>namedDest</i> | The named destination associated with this destination. If this is non- <i>NULL</i> , the other attributes are ignored. This destination may contain multibyte characters. |
| <i>nameLength</i> | Length of <i>namedDest</i> , in bytes. |
| <i>pageNum</i> | The page number of the destination view. |
| <i>fitType</i> | The fit type of the destination view. Must be one of View Destination Fit Types . |

destRect

A rectangle enclosing the destination view.

zoom

The zoom factor of the destination view. Use zero to inherit the zoom.

AVDocOpenParams

```
typedef struct _t_AVDocOpenParams {
    ASSize_t size;
    ASBool useFrame;
    AVRect frame;
    ASBool useVisible;
    ASBool visible;
    /* Available only in or after Acrobat 3.0 */
    ASBool useServerType;
    const char* serverType;
    void* serverCreationData;
    ASBool useSourceDoc;
    AVDoc sourceDoc;
    ASBool useReadOnly;
    ASBool readOnly;
    ASBool useViewType;
    const char* viewType;
    ASBool useViewDef;
    AVDocViewDef viewDef;
} AVDocOpenParamsRec, *AVDocOpenParams;
```

Description

Parameters used when opening a file using [AVDocIsReadOnly](#), [AVDocOpenFromFileWithParams](#), or [AVDocOpenFromPDDocWithParams](#).

In UNIX, it is not possible to set the frame of the *NULL* document (that is, the window to show when no document is open) using this data structure.

Header File

AVExpT.h

Related Methods

[AVDocIsReadOnly](#)
[AVDocOpenFromFileWithParams](#)
[AVDocOpenFromPDDocWithParams](#)

size

Size of the data structure. Must be set to *sizeof(AVDocOpenParamsRec)*.

useFrame

If *true*, *frame* is used to specify the size and location of the window into which the document is opened. If *false*, *frame* is ignored and the default frame is used instead. See also *visible*.

frame

An [AVRect](#) specifying the size and location of the window into which the document is opened.

In Mac OS, the coordinates are global screen coordinates.

In Windows, the coordinates are MDI client coordinates.

See also *visible*.

useVisible

If *true*, *visible* is used to determine whether or not the window is visible after the document is opened. If *false*, *visible* is ignored. See also *visible*.

visible

Specifies the window's visibility. If *visible* is *false* and *useVisible* is *true*, *frame* is ignored—regardless of the setting of *useFrame*.

In Mac OS, if *true*, the document is opened into a visible window. If *false*, the document is opened into a hidden window.

In Windows, if *true*, the document is opened in a visible window. If *false*, the document is opened in a minimized window.

useServerType

Indicates whether the *serverType* and *serverCreationData* fields are used.

serverType

Name of AVDoc server for this [AVDoc](#):

"EXTERNAL" — the *AVDoc* server for an external window

serverCreationData

Platform-dependent server data to associate with the *AVDoc* server. For a *serverType* of "EXTERNAL", must be of type [ExternalDocServerCreationData](#).

useSourceDoc

Indicates whether the *sourceDoc* field is used.

sourceDoc

[AVDoc](#) whose window will be taken over by new document. *sourceDoc* will be closed at the same time.

useReadOnly

Indicates whether the *readOnly* field is used.

readOnly

If *true*, open the document in read-only mode.

useViewType

Indicates whether the *viewType* field is used.

viewType

Type of view to open on document. Permissible values are:

"*AVPageView*" — Displays only the *AVPageView*, that is, the window that displays the PDF file. Does not display scrollbars, the toolbar, and bookmark or thumbnails pane. Annotations, such as links, are active.

"*AVDocView*" — Displays the *AVPageView*, scrollbars, and bookmark or thumbnails pane. Annotations, such as links, are active.

"*AVExternalView*" — Displays the *AVPageView*, scrollbars, toolbar, and bookmark or thumbnails pane. Annotations, such as links, are active.

"*AVEmbeddedView*" — Embeds the PDF file in an external document, an HTML file, for example. Show the first page of the PDF file; no scrollbars, the toolbar, and bookmark or thumbnails pane are visible. Annotations, such as links, are *not* active.

useViewDef

Indicates whether the *viewDef* field is used.

viewDef

Initial view with which to open the document. Must be an [*AVDocViewDef*](#).

AVDocPrintParams

```
typedef struct _t_AVDocPrintParams {
    ASSize_t size;
    ASBool interactive;
    ASBool cancelDialog;
    ASInt32 firstPage;
    ASInt32 lastPage;
    ASInt32 psLevel;
    ASBool binaryOK;
    ASBool shrinkToFit;
    ASAtom fileSysName;
    ASPathName filePathName;
    ASPlatformPrinterSpec printerSpec;
    ASBool embedded;
    AVRect32 embeddedRect;
    ASBool emitToPrinter;
    ASBool emitToFile;
    ASBool doColorSeparations;
    ASEnum8 emitFileOption;
    ASEnum8 emitFontOption;
    ASUns32 emitFlags;
    /* New in Acrobat 4.0 */
    PDPageRange* ranges;
    ASInt16 numRanges;
    ASBool TTasT42;
    ASBool printAsImage;
    ASBool printerHasFarEastFonts;
    ASBool reverse;
    ASInt32 pageSpec;
} AVDocPrintParamsRec, *AVDocPrintParams;
```

| | |
|------------------------|---|
| Description | Structure that specifies how to print a document. |
| Header File | <i>AVExpT.h</i> |
| Related Types | EmitFontOptions Emit Flags |
| Related Methods | AVDocPrintPagesWithParams |

size

Size of the data structure. Must be set to *sizeof(AVDocPrintParamsRec)*.

One and only one of the following booleans must be set:

- *interactive* — Displays a Print dialog box and print status window while printing.
- *cancelDialog* — Displays a Cancel dialog box while printing.
- *embedded* — Renders one page scaled to the size specified by *embeddedRect*.
- *emitToPrinter* — Noninteractive output to a printer without a Print dialog box or status window.
- *emitToFile* — Noninteractive output to a file. Used to emit color separations or EPS. This flag *cannot* be used with the Acrobat Reader.

interactive

If *true*, displays dialog boxes; otherwise does not display dialog boxes.

cancelDialog

If *interactive* is *false* and *cancelDialog* is *true*, a Cancel dialog box appears.

firstPage, *lastPage*, *psLevel*, *binaryOk*, and *shrinkToFit* are used if *emitToPrinter* or *emitToFile* are *true*.

firstPage

First page to print. The first page is 0. If -1, all pages are printed.

lastPage

Last page to print. If *firstPage* is -1, this parameter is ignored.

psLevel

If printing to PostScript, 1 means emit as level 1, 2 means level 2.

binaryOK

true if a binary channel to the printer is supported, *false* otherwise.

shrinkToFit

true if the page is scaled to fit the printer page size, *false* otherwise.

fileSysName and *filePathName* are used if *emitToPrinter* or *emitToFile* is *true*.

If *emitToPrinter* is *true* and *filePathName* is non-NULL, the system printer driver is used to emit the output stream to the file. Implemented for Windows only.

fileSysName

The file system name; see *filePathName*.

For the operation of printing to a printer (*emitToPrinter* = *true*), if *filePathName* is specified, *fileSysName* must be the name of the default file system. You can get the file system's name from the [ASFileSysGetFileSysNameProc](#) callback in the [ASFileSysRec](#) of the file system you are using.

filePathName

If non-*NULL*, *filePathName* is a platform path for the specified *fileSysName*, or, if *fileSysName* is *ASAtomNull*, it is one of the following:

- In Windows: a C-string path name
- In Mac OS: a *FSSpecPtr*
- In UNIX: a C-string path name

printerSpec

Optionally used if *interactive*, *embedded*, or *emitToPrinter* is *true*. If *NULL*, a default system printer is used. If non-*NULL*, *printerSpec* is a platform-specific value. Must be an [ASPlatformPrinterSpec](#).

embedded

true if an embedded view of a page to print, *false* otherwise.

- *firstPage* and *lastPage* must be the same.
- *embeddedRect* specifies the location on the page where the view of the page is to appear.
- The printer must be specified as an *HDC* or *CGrafPtr*.

embeddedRect

Location on the page where the view of the page is to appear, specified in device coordinates for the current printer.

emitToPrinter

If *true*, use the system printer driver for output. If *filePathName* is specified, uses the driver to create the file.

The following parameters are for emission of PostScript Level 1 Color Separations and EPS—or standard PostScript. Creates and writes to *filePathName* (may not be *NULL*). Does not use the system printer driver. Only has partial font emitting capabilities on some platforms:

- Macintosh: embedded and system Type 1 fonts only; no TrueType or substitution fonts
- UNIX: all fonts
- Windows: embedded and system Type 1 fonts only; no TrueType or substitution fonts

emitToFile

If *true*, emit non-interactive output to a file. Used to emit color separations or EPS. This flag cannot be used with the Acrobat Reader.

doColorSeparations

Perform level 1 color separations. See *Color Separation Conventions for PostScript Language Programs*, Technical Note #5044.

emitFileOption

File output options: PostScript or EPS, with or without a preview. *Must* be one of the following values:

kAVEmitFilePS — PostScript file

kAVEmitFileEPSNoPrev — EPS file with no preview

kAVEmitFileEPSMacStdPrev — EPS file with standard preview

kAVEmitFileEPSMacExtPrev — EPS file with extended preview

emitFontOption

Font output options. Must be one of [EmitFontOptions](#).

emitFlags

Additional emit options. Must be one of [Emit Flags](#).

The following parameters provide support for multiple page ranges.

ranges

Must be a [PDPageRange](#).

numRanges

Range of pages to print.

The following parameters control TrueType --> Type 1 conversion for PostScript printing.

TTasT42

If *true*, send TrueType fonts as TrueType fonts (level 3 and most level 2 PostScript printers. If *false*, convert TrueType to Type 1. This is typically *only* desirable for Level 1 PostScript where no TrueType handling is present.

printAsImage

If *true*, print pages as an image.

printerHasFarEastFonts

If *true*, do not down load Far East fonts to printer.

reverse

Print from *lastPage* to *firstPage*.

pageSpec

Pages in the range to print. Must be one of: *PDAllPages*, *PDEvenPagesOnly*, or *PDOddPagesOnly*. See [Page Specification](#).

AVDocSaveParams

```
typedef struct _t_AVDocSaveParams {  
    ASSize_t size;  
    ASBool useSaveDialog;  
} AVDocSaveParamsRec, *AVDocSaveParams;
```

Description Structure used by [AVDocDoSaveAsWithParams](#) containing parameters that a client wishing to save a file might want to specify.

Header File *AVExpt.h*

Related Callbacks None

Related Methods [AVDocDoSaveAsWithParams](#)

| |
|---|
| <i>size</i> Size of the data structure. Must be set to <i>sizeof(AVDocSaveParamsRec)</i> . |
|---|

| |
|--|
| <i>ASBooluseSaveDialog</i> Use the standard file Save dialog box. |
|--|

AVDocSelectionServer

```
typedef struct _t_AVDocSelectionServer {
    ASSize_t size;
    AVDocSelectionGetTypeProc GetType;
    AVDocSelectionGettingSelectionProc GettingSelection;
    AVDocSelectionAddedToSelectionProc AddedToSelection;
    AVDocSelectionLosingSelectionProc LosingSelection;
    AVDocSelectionRemovedFromSelectionProc RemovedFromSelection;
    AVDocSelectionCanSelectAllProc CanSelectAll;
    AVDocSelectionSelectAllProc SelectAll;
    AVDocSelectionCanPropertiesProc CanProperties;
    AVDocSelectionPropertiesProc Properties;
    AVDocSelectionCanDeleteProc CanDelete;
    AVDocSelectionDeleteProc Delete;
    AVDocSelectionCanCopyProc CanCopy;
    AVDocSelectionCopyProc Copy;
    AVDocSelectionEnumSelectionProc EnumSelection;
    AVDocSelectionShowSelectionProc ShowSelection;
    AVDocSelectionCanCutProc CanCut;
    AVDocSelectionCutProc Cut;
    AVDocSelectionCanPasteProc CanPaste;
    AVDocSelectionPasteProc Paste;
    AVDocSelectionKeyDownProc KeyDown;
    /* Renamed in Acrobat 4.0--was AVDocHighlightSelectionProc */
    AVDocSelectionHighlightSelectionProc HighlightSelection;
    /* New in Acrobat 4.0 */
    AVDocSelectionGetSelectionTypeProc GetSelectionType;
    AVDocSelectionEnumPageRangesProc EnumPageRanges;
} AVDocSelectionServerRec, *AVDocSelectionServer;
```

Description Data structure containing callbacks that implement a selection server. The callbacks implement the selection server functions. For example, add an item to the selection, remove an item from the selection, copy the current selection to the clipboard.

Header File *AVExpT.h*

Related Methods [AVDocRegisterSelectionServer](#)
[AVDocGetSelectionServerByType](#)

size

Size of the data structure. Must be set to `sizeof(AVDocSelectionServerRec)`.

AVDocViewDef

```
typedef struct _t_AVDocViewDef {
    ASSize_t size;
    ASBool bringToFront;
    ASBool usePageViewInfo; /* pageview info */
    PDLayoutMode pageViewLayoutMode;
    ASInt32 pageViewPageNum;
    AVZoomType pageViewZoomType;
    ASFixed pageViewZoom;
    ASInt16 pageViewX;
    ASInt16 pageViewY;
    ASBool pageViewStartThread;
    ASInt32 pageViewThreadIndex;
    PDBead pageViewBead;
    ASBool useOverViewInfo; /* overview info */
    PDPageMode overViewMode;
    ASInt16 overViewPos;
    ASInt32 overViewX;
    ASInt32 overViewY;
    ASBool useWindowInfo; /* window info */
    AVRect windowFrame;
    ASBool unused1;
    const char* unused2;
} AVDocViewDefRec, *AVDocViewDef;
```

| | |
|------------------------|---|
| Description | Structure that defines a view of a document, including page, zoom, and so on. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocGetViewDef |

| |
|---|
| <i>size</i> |
| Size of the data structure. Must be set to <i>sizeof(AVDocViewDef)</i> . |
| <i>bringToFront</i> |
| If <i>true</i> , bring window to front; if <i>false</i> , don't bring window to front. |
| <i>usePageViewInfo</i> |
| If <i>true</i> , use the next 6 page view fields. If <i>false</i> , it does not use them. |
| <i>pageViewLayoutMode</i> |
| Page layout mode; must be one of PDLayoutMode . |

| | |
|----------------------------|--|
| <i>pageViewPageNum</i> | Page number. |
| <i>pageViewZoomType</i> | Zoom type; must be one of AVZoomType . |
| <i>pageViewZoom</i> | Zoom factor; used if <i>pageViewZoomType</i> is <i>AVZoomNoVary</i> . Use zero to inherit zoom. |
| <i>pageViewX</i> | The x-coordinate to scroll to. |
| <i>pageViewY</i> | The y-coordinate to scroll to. |
| <i>pageViewStartThread</i> | If <i>true</i> , use the next two article thread fields. If <i>false</i> , it does not use them. |
| <i>pageViewThreadIndex</i> | Current thread index. |
| <i>pageViewBead</i> | Current PDBead . |
| <i>useOverViewInfo</i> | If <i>true</i> , use the next four view fields. If <i>false</i> , it does not use them. |
| <i>overViewMode</i> | The PDPageMode to use. |
| <i>overViewPos</i> | Position of splitter. |
| <i>overViewX</i> | The x-coordinate to scroll to in bookmark or thumbnail pane. |
| <i>overViewY</i> | The y-coordinate to scroll to in bookmark or thumbnail pane. |
| <i>useWindowInfo</i> | If <i>true</i> , use the <i>windowFrame</i> field. If <i>false</i> , it does not use them. |
| <i>windowFrame</i> | New window frame in which to display the document. |

unused1

Currently unused.

unused2

Currently unused.

AVIcon

```
#define AVIcon void*
```

| | |
|--------------------------|--|
| Description | An icon on a menu item or tool button. |
| Header File | <i>AVCalls.h</i> |
| Related Callbacks | None |
| Related Methods | <u>AVMenuItemNew</u> <u>AVToolButtonGetIcon</u> <u>AVToolButtonNew</u> <u>AVToolButtonSetIcon</u> |

AVPageViewControllerID

```
typedef AEnum16 AVPageViewControllerID;
enum {
    kAVPageViewZoomControl,
    kAVPageViewPageFlipControls,
    kAVPageViewPageNumberControl,
    kAVPageViewPageSizeControl,
    kAVPageViewSplitterBar,
    kAVPageViewHorizontalScrollBar,
    kAVPageViewVerticalScrollBar
};
```

| | |
|--------------------------|--|
| Description | Used with AVPageViewShowControl to allow a plug-in author to turn on and off the controls shown in the status area at the bottom of a page view. |
| Header File | <i>AVExpt.h</i> |
| Related Callbacks | None |
| Related Methods | AVPageViewShowControl |

| |
|---------------------------------------|
| <i>kAVPageViewZoomControl</i> |
| The zoom control. |
| <i>kAVPageViewPageFlipControls</i> |
| The page flip control. |
| <i>kAVPageViewPageNumberControl</i> |
| The page number control. |
| <i>kAVPageViewPageSizeControl</i> |
| The page size control. |
| <i>kAVPageViewSplitterBar</i> |
| The splitter bar control. |
| <i>kAVPageViewHorizontalScrollBar</i> |
| The horizontal scroll bar control. |
| <i>kAVPageViewVerticalScrollBar</i> |
| The vertical scroll bar control. |

AVPrefsType

| | |
|------------------------|--|
| Description | Enumerated data type containing the Acrobat viewer's preferences settings. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVAppGetPreference AVAppSetPreference |

avpAllowOpenFile — *ASBool*

If *true*, links or actions that might launch another application are allowed. If *false*, such links do nothing. Corresponds to the "Allow Open File" item in the General Preferences dialog box. See *avpSecureOpenFile*.

avpAntialiasGraphics — *ASBool*

Turns on or off anti-aliasing of graphics. Users can control the anti-aliasing of images and text from General Preferences with the Smooth Text & Images checkbox

avpAntialiasLevel — *ASInt16* (Present only in version 3.0 and higher)

Point size above which to use anti-aliasing.

avpAntialiasText — *ASBool* (Present only in version 3.0 and higher)

Turns on or off anti-aliasing, that is, text smoothing.

avpASExtensionDigCert — *ASExtensionEncryptedDigitalCertificateRec*

Currently unused.

avpBookmarkShowLocation — *ASBool*

If *true*, the bookmarks corresponding to the portion of the document currently being viewed are displayed in bold. If *false*, it is not.

avpBrowserIntegration — *ASBool* (Windows only)

If *true*, the *PDFViewer* plug-in is used to view PDF files opened inside a browser window. If *false*, the viewer is launched as a helper application to view the file.

avpCaseSensitive — *ASBool*

If *true*, the Acrobat viewer's Find command (not the Search plug-in) performs case-sensitive searches. If *false*, searches are not case-sensitive.

| |
|---|
| <i>avpCurrCMM</i> — <i>char*</i> |
| Currently unsupported. Tracks what the current CMM is, for example, Apple ColorSync, Kodak, and so forth. Users can control this with the Color Manager part of the General Preferences dialog box. |
| <i>avpDefaultOverviewType</i> — <i>ASInt32</i> |
| Whether thumbnail images, bookmarks, or neither should be shown along with documents by default. Must be one of the PDPageMode values. |
| <i>avpDefaultSplitterPos</i> — <i>ASInt32</i> |
| The default width (in pixels) of the portion of the document window in which bookmarks and thumbnail images are displayed. The actual width can be changed by the user or programmatically using AVDocSetSplitterPosition . |
| <i>avpDefaultZoomScale</i> — <i>ASFixed</i> |
| Default magnification when a document is opened. |
| <i>avpDefaultZoomType</i> — AVZoomType |
| Default zoom type for a page view. Must be one of the AVZoomType values. |
| <i>avpDestFitType</i> — <i>char*</i> |
| Currently unsupported. Default destination fit type used for bookmark and link creation. Must be one of the View Destination Fit Types . |
| <i>avpDestZoomInherit</i> — <i>ASBool</i> |
| If <i>true</i> , the default destination fit type for creating new links and bookmarks is "Inherit Zoom." If <i>false</i> , it is not. |
| <i>avpDoCalibratedColor</i> — <i>ASBool</i> |
| If <i>true</i> , the Acrobat viewer renders using calibrated color. The chromaticities and gammas used are those set using PDPrefSetColorCal . If <i>false</i> , do not. |
| <i>avpDownloadEntireFile</i> — <i>ASBool</i> (Present only in version 3.0 and higher) |
| If <i>true</i> , download entire PDF files in browser windows. If <i>false</i> , do not. |
| <i>avpEmitHalftones</i> — <i>ASBool</i> (Present only in version 3.0 and higher) |
| If <i>true</i> , use halftones found in PDF files. If <i>false</i> , do not. |

| |
|---|
| <i>avpGreekLevel</i> — <i>ASInt32</i> |
| Size, in points, below which text is greeked if <i>avpGreekText</i> is <i>true</i> . |
| <i>avpGreekText</i> — <i>ASBool</i> |
| If <i>true</i> , text smaller than <i>avpGreekLevel</i> is greeked (displayed as gray boxes). If <i>false</i> , all text is drawn, regardless of its size. |
| <i>avpHighlightColor</i> — <i>PDColorValueRec</i> |
| Currently unsupported. |
| <i>avpHighlightMode</i> — <i>ASInt32</i> (<i>Used on Macintosh only</i>) |
| Specifies the way in which highlighted text is displayed. Must be one of the following: <i>HIGHLIGHT_PAINT_XOR</i> — Paint highlight color in XOR mode <i>HIGHLIGHT_INVERT_MAC</i> — Invert in special Macintosh highlight mode <i>HIGHLIGHT_INVERT_XOR</i> — Invert in XOR mode This preference exists because the standard Macintosh highlighting generally works quite well, but can occasionally become invisible when text is on a colored background. |
| <i>avpIgnorePageClip</i> — <i>ASBool</i> |
| If <i>true</i> , the viewer ignores clipping paths that would cause a thin white border to appear around the edges of the page. If <i>false</i> , the viewer honors all clipping paths. |
| <i>avpMarkHiddenPages</i> — <i>ASBool</i> (<i>Present only in version 3.0 and higher</i>) |
| Mark hidden (for Presenter plug-in) pages with the appropriate mark. |
| <i>avpMaxOpenDocuments</i> — <i>ASInt32</i> (<i>Present only in version 3.0 and higher</i>) |
| Maximum number of open documents allowed. |
| <i>avpMaxThreadZoom</i> — <i>ASFixed</i> |
| The maximum zoom factor that is automatically used when the Acrobat viewer enters "Follow Article" mode. A value of 1.0 corresponds to a zoom factor of 100%. |
| <i>avpMinimizeBookmarks</i> — <i>ASBool</i> |
| If <i>true</i> , hide or minimize the Bookmarks panel after the user has selected a bookmark. If <i>false</i> , do not. |
| <i>avpNoteColor</i> — PDColorValue |
| Default color to use for new notes. |

| |
|--|
| <i>avpNoteFontName</i> — <i>char*</i> (Present only in version 3.0 and higher) Currently unsupported. Font name for text notes. |
| <i>avpNoteFontSize</i> — <i>ASInt32</i> Font size for text notes. |
| <i>avpNoteLabel</i> — <i>char*</i> Currently unsupported. Default label to use for new notes. |
| <i>avpNoteLabelEncoding</i> — <i>void*</i> Indicates the script code (Mac OS) or charset (Windows) that the <i>avpNoteLabel</i> is encoded in. In both Mac OS and Windows, these codes are integers even though the preference is typed as <i>void*</i> . |
| <i>avpOpenDialogAtStartup</i> — <i>ASBool</i> If <i>true</i> , the File Open dialog box is displayed when the Acrobat viewer is launched without a document to open. If <i>false</i> , it is not. |
| <i>avpOpenInPlace</i> — <i>ASBool</i> If <i>true</i> , cross-document links should open in the same window. If <i>false</i> , open the file in another window. |
| <i>avpPageUnits</i> — <i>PageUnitsType</i> Current units (0 = points, 1 = inches, 2 = millimeters). |
| <i>avpPageViewLayoutMode</i> — PDLayoutMode Default layout mode. |
| <i>avpPrefsVersion</i> — <i>ASInt32</i> (Read only) The preferences file format version number. |
| <i>avpPrintAnnots</i> — <i>ASBool</i> If <i>true</i> , prints annotations on the page (useful for summarize annotations that the user can control from the Print dialog box). If <i>false</i> , non-Forms annotations with appearances (that is, mark-up annotations) will never print. |
| <i>avpPSLevel</i> — <i>ASInt32</i> The PostScript language level to use when printing to a PostScript printer. Valid values are 1 and 2. |

*avpRecentFile1, avpRecentFile2, avpRecentFile3, avpRecentFile4 — char**
(Present only in version 3.0 and higher)

Currently unsupported.

Paths for most recently used files in recent file menu.

avpRememberDialogs — ASBool

If *true*, the Acrobat viewer remembers the location of certain dialog boxes (such as the Find dialog box) and displays them in their previous location. If *false*, they are displayed in a default location.

avpSaveAsLinearized — ASBool

If *true*, save as linearized file when saving. If *false*, do not save as linearized.

avpSecureOpenFile — ASBool

If *true*, links or actions that might launch another application are not allowed. A plug-in can set this preference to *true* to override the *avpAllowOpenFile* preference (that the user can control from the General Preferences dialog box). If *false*, such links are allowed.

avpSendFarEastFonts — ASBool

If *true*, Type0 fonts used in the PDF file are included in the PostScript file that is generated. If *false*, the fonts are not be downloaded and the printer must already have the appropriate font.

avpShortMenus — ASBool (Ignored in Acrobat 3.0 or later)

If *true*, the Acrobat viewer displays short menus. If *false*, it displays long menus. See [AVMenuItemNew](#) for information on specifying whether a menu item should be visible only when long menus are shown.

avpShowAnnotSequence — ASBool

If *true*, displays and prints annotation sequence numbers on annotations (useful for summarize annotations that the user can control from the Annotations Preferences dialog box). If *false*, do not.

avpShowHiddenAnnots — ASBool (Present only in version 3.0 and higher)

If *true*, shows the annotations marked as hidden. If *false*, such annotations are not displayed.

avpShowLargeImages — ASBool

If *true*, the Acrobat viewer displays large images. If *false*, gray boxes are shown in place of large images, reducing rendering time for pages with large images.

avpShowLeftToolBar — ASBool

Indicates whether the left toolbar is visible or invisible.

In Windows, this controls the visibility of the toolbar at application launch. In Mac OS, this controls the visibility of the toolbar at the time a document is opened.

avpShowMenuBar — ASBool

Indicates whether the menu bar is visible or invisible when the application first launches.

avpShowSplashAtStartup — ASBool

If *true*, the Acrobat viewer splash screen is shown when the Acrobat viewer is launched. If *false*, it is not.

avpShowToolBar — ASBool

If *true*, the Acrobat viewer's toolbar is displayed. If *false*, it is not. The toolbar can also be shown and hidden by the user.

avpShrinkToFit — ASBool

If *true*, pages are shrunk to fit the imageable area of the printer when printed. If *false*, pages are not shrunk to fit.

avpSkipWarnings — ASBool

If *true*, a warning dialog box is not displayed when a user deletes notes, bookmarks, links, pages, or thumbnails. If *false*, the dialog box is displayed.

avpSubstituteFontType — ASInt32

Determines whether sans serif, serif, or both substitution fonts are available when printing. Using only one substitution font type generally reduces the quality of font substitution, but may allow some files that require font substitution to print on PostScript printers that have very little memory. Valid values are:

- 0 — Use both sans serif and serif
- 1 — Use sans serif only
- 2 — Use serif only

avpSuppressCSA — ASBool

Used by the PostScript print code to determine whether to emit color space arrays for 4- or 1-component ICCBased color spaces. If *true*, do not emit CSAs, instead emit **DefaultCMYK** or **DefaultGray**, respectively. The user may control this in Acrobat via File | DocInfo | Prepress.

| |
|--|
| <i>avpThumbViewScale</i> — <i>ASFixed</i> |
| The scale at which thumbnail images are created. The Acrobat viewer's default is <i>fixedEighth</i> , creating thumbnail images whose linear dimensions are one-eighth those of the actual page. |
| <i>avpThumbViewTimeout</i> — <i>ASInt32</i> |
| Currently unsupported. |
| <i>avpUseHostFont</i> — <i>ASBool</i> |
| Currently unsupported. |
| <i>avpUseLocalFonts</i> — <i>ASBool</i> |
| If <i>true</i> , the viewer uses fonts installed on the user's system when displaying and printing documents. If <i>false</i> , the viewer only uses its local copy of the base 14 fonts when displaying documents. All other fonts will be fauxed or substituted. |
| <i>avpUseLogicalPageNumbers</i> — <i>ASBool</i> |
| If <i>true</i> , the page numbering information encoded in the document is used. If <i>false</i> , the viewer numbers pages using decimal numbers starting at 1. |
| <i>avpWholeWords</i> — <i>ASBool</i> |
| If <i>true</i> , the Acrobat viewer's Find command (not the Search plug-in) matches only whole words. If <i>false</i> , the partial words are also matched. |

Table 1 Full-Screen Mode Preferences

| |
|--|
| <i>avpFullScreenChangeTimeDelay</i> — <i>ASInt32</i> |
| Time (in seconds) to show each page when using automatic page changing in full-screen mode. |
| <i>avpFullScreenClick</i> — <i>ASBool</i> (<i>Present only in version 3.0 and higher</i>) |
| <i>true</i> if the page advances on a mouse click in full-screen mode, <i>false</i> otherwise. |
| <i>avpFullScreenColor</i> — PDColorValue |
| The background color to use when the Acrobat viewer is in full-screen mode. |
| <i>avpFullScreenCursor</i> — <i>ASInt16</i> (<i>Present only in version 3.0 and higher</i>) |
| Cursor behavior in full-screen mode (0 = visible, 1 = hidden, 2 = hidden after delay). |
| <i>avpFullScreenEscape</i> — <i>ASBool</i> (<i>Present only in version 3.0 and higher</i>) |
| <i>true</i> if the Escape key exits full-screen mode, <i>false</i> otherwise. |

Table 1 Full-Screen Mode Preferences

| |
|---|
| <p><i>avpFullScreenLoop</i> — <i>ASBool</i></p> <p><i>true</i> if the document's pages are displayed in a loop (rather than just once) when using full-screen mode, <i>false</i> otherwise.</p> |
| <p><i>avpFullScreenTransitionType</i> — <i>char*</i> (Present only in version 3.0 and higher)</p> <p>Default transition name.</p> |
| <p><i>avpFullScreenUsePageTiming</i> — <i>ASBool</i> (Present only in version 3.0 and higher)</p> <p><i>true</i> if page timings are used to advance pages in full-screen mode, <i>false</i> otherwise.</p> |
| <p><i>avpFullScreenUseTimer</i> — <i>ASBool</i> (Present only in version 3.0 and higher)</p> <p><i>true</i> if the page advances automatically in full-screen mode, <i>false</i> otherwise.</p> |

Table 2 Caching Preferences

| |
|---|
| <p><i>avpEnablePageCache</i> — <i>ASBool</i></p> <p><i>true</i> if the following page is rendered offscreen while the current page is viewed, improving performance when a document is viewed sequentially, <i>false</i> if no draw-ahead is used.</p> |
| <p><i>avpMaxCosDocCache</i> — <i>ASInt32</i></p> <p>The maximum zoom factor at which pages will be cached. Pages viewed at a higher zoom factor will not be cached. A value of 1.0 corresponds to a zoom factor of 100%.</p> <p><i>Note: A platform-dependent minimum value for the cos cache is enforced. It is 2M for Windows and Unix and 500K for Mac. If a plug-in attempts to set the maxCosDocCache below the minimum, the plug-in's setting is ignored and the minimum is used instead.</i></p> |
| <p><i>avpMaxPageCacheBytes</i> — <i>ASInt32</i></p> <p>The maximum number of bytes the page cache is allowed to occupy.</p> |
| <p><i>avpMaxPageCacheZoom</i> — <i>ASFixed</i></p> <p>The maximum zoom factor at which pages will be cached. Pages viewed at a higher zoom factor will not be cached. A value of 1.0 corresponds to a zoom factor of 100%.</p> |
| <p><i>avpMinPageCacheTicks</i> — <i>ASInt32</i></p> <p>The minimum number of ticks needed to redraw a page before it will be cached. Pages that can be redrawn in less time will not be cached. A tick is 1/60 of a second.</p> |

Table 2 Caching Preferences

| |
|--|
| <i>avpPersistentCacheFolder</i> — ASPathName Location of the Acrobat cache, which is used while viewing PDF files on a slow file system. |
| <i>avpPersistentCacheSize</i> — <i>ASInt32 (Present only in version 3.0 and higher)</i> Size of the Acrobat cache, which is used while viewing PDF files on a slow file system. |

AVRect

AVRectP

```
typedef struct _t_AVRect {
    ASInt16 left;
    ASInt16 top;
    ASInt16 right;
    ASInt16 bottom;
} AVRect, *AVRectP;
```

| | |
|------------------------|---|
| Description | <p>Data structure representing a rectangle (a quadrilateral having only horizontal and vertical sides).</p> <p>The AcroView coordinate system is defined so that (0,0) is at the top, x increases to the right, and y increases down (the same as GDI and QuickDraw but opposite to the PostScript language). An <i>AVRect</i> is defined so that its <i>top</i> is above its <i>bottom</i>, but this means that $0 < \text{top} < \text{bottom}$.</p> |
| Header File | <i>AVExpT.h</i> |
| Related Methods | Numerous |

AVRect32

AVRect32P

```
typedef struct _t_AVRect32 {
    ASInt32 left;
    ASInt32 top;
    ASInt32 right;
    ASInt32 bottom;
} AVRect32, *AVRect32P;
```

| | |
|------------------------|---|
| Description | <p>Data structure representing a rectangle (a quadrilateral having only horizontal and vertical sides).</p> <p>The AcroView coordinate system is defined so that (0,0) is at the top, x increases to the right, and y increases down (the same as GDI and QuickDraw but opposite to the PostScript language). An <i>AVRect32</i> is defined so that its top is above its bottom, but this means that $0 < \text{top} < \text{bottom}$.</p> |
| Header File | <i>AVExpT.h</i> |
| Related Methods | Numerous |

AVSystemFont

```
typedef struct _t_AVSystemFont {  
    short fondID;  
    short styleID;  
    ASUns32 flags;  
    ASAtom pdfFontName;  
} AVSystemFontRec, *AVSystemFont;
```

Description *(Macintosh only, present only in version 3.0 or later)*
System font.

Header File *AVExpT.h*

Related Methods [AVAppEnumSystemFonts](#)

fondID

Macintosh FOND id.

styleID

Macintosh style value: normal, italic, bold or bold | italic.

flags

Font flags. Must be one of [AVSystemFont Flags](#).

pdfFontName

PostScript name or TrueType styled name.

AVSystemFont Flags

| | |
|------------------------|--|
| Description | Constants that are used to specify the attributes of an AVSystemFont . |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVAppEnumSystemFonts |

| | |
|----------------------------|---|
| <i>kFontIsType1</i> | Type 1 font. |
| <i>kFontIsMMMaster</i> | Multiple Master font. |
| <i>kFontIsMMInstance</i> | Multiple Master font (completely-specified instance). |
| <i>kFontIsTrueType</i> | True type font. |
| <i>kFontIsCIDFontType0</i> | Character ID Type 0 font. |
| <i>kFontIsOCFType1</i> | OCF Type 1 font. |

AVTool

```
typedef struct _t_AVTool {
    ASSize_t size;
    ActivateProcType Activate;
    DeactivateProcType Deactivate;
    DoClickProcType DoClick;
    AdjustCursorProcType AdjustCursor;
    DoKeyDownProcType DoKeyDown;
    GetTypeProcType GetType;
    IsPersistentProcType IsPersistent;
    ASInt32 cursorID;
    AVComputeEnabledProc ComputeEnabled;
    void* computeEnabledData;
    /* Added in Acrobat 4.0 */
    DoClickProcType DoRightClick;
    DoLeaveProcType DoLeave;
    GetSelectionServerProcType GetSelectionServer;
} AVToolRec, *AVTool;
```

Description Data structure for a tool. It contains callbacks that implement the tool's functions (for example, handling mouse clicks and controlling the cursor shape).

Header File *AVExpT.h*

Related Methods [AVAppGetActiveTool](#)
[AVAppSetActiveTool](#)
[AVAppGetDefaultTool](#)
[AVAppGetLastActiveTool](#)
[AVAppGetToolByName](#)
[AVAppRegisterTool](#)
[AVToolGetType](#)
[AVToolsIsPersistent](#)

size

Size of the data structure. Must be set to *sizeof(AVToolRec)*.

cursorID

A default cursor, used if the tool does not have an [AdjustCursorProcType](#) callback.

computeEnabledData

User-supplied data to pass to the tool's [AVComputeEnabledProc](#) callback each time it is called.

AVTransHandler

```
typedef struct _t_AVTransHandler {
    ASSize_t size;
    AVTransHandlerGetTypeProc GetType;
    AVTransHandlerExecuteProc Execute;
    AVTransHandlerGetUINameProc GetUIName;
    AVTransHandlerGetItemUINameProc GetItemUIName;
    AVTransHandlerInitTransDictProc InitTransDict;
    AVTransHandlerCompleteTransDictProc CompleteTransDict;
    /* The following callbacks are for non-standard plug-ins that
       have additional information */
    AVTransHandlerDoPropertiesProc DoProperties;
    AVTransHandlerGetInstructionsProc GetInstructions;
    AVTransHandlerGetButtonTextProc GetButtonText;
    AVTransHandlerGetStringOneTextProc GetStringOneText;
    AVTransHandlerGetStringTwoTextProc GetStringTwoText;
} AVTransHandlerRec, *AVTransHandler;
```

Description Data structure containing callbacks that implement a transition handler. The callbacks implement the transition handler functions.

Header File *AVExpT.h*

Related Methods [AVAppRegisterTransHandler](#)

size

Size of the data structure. Must be set to *sizeof(AVTransHandlerRec)*.

AVTransitionPort

```
typedef struct _t_AVTransitionPort {
    void reserved1;
    OPAQUE_32_BITS reserved2;
    OPAQUE_32_BITS reserved3;
    OPAQUE_32_BITS reserved4;
    OPAQUE_32_BITS reserved5;

    /* The following two fields vary by platform */

    #if WIN_PLATFORM
        HDC hdc;
        RECT rect;

    #elif MAC_PLATFORM
        GWorldPtr port;
        Rect rect;

    #elif UNIX_PLATFORM
        void* port;
        void* rect;

    #endif
} AVTransitionPortRec *AVTransitionPort;
```

Description

Platform-dependent data structure for a transition.

In general, a transition port specifies a bitmap and a rectangle describing the portion of the bitmap affected by the transition. The transition handler's [AVTransHandlerExecuteProc](#) callback must copy all the bits from the source port's bitmap within the source port's rectangle to the area in the destination port's bitmap described by the destination port's rectangle. The source and destination ports' rectangles are guaranteed to be equal in size.

Header File

AVExpT.h

Related Callbacks

[AVTransHandlerExecuteProc](#)

AVWindow Flags

| | |
|------------------------|---|
| Description | Constants that are used to specify the attributes of an <i>AVWindow</i> when it is created. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVWindowNew AVWindowNewFromPlatformThing |

| | |
|-----------------------------|---|
| <i>AVWIN_RESIZEABLE</i> | Window is resizable. UNIX — Under OpenWindows, all windows have resize corners regardless of the setting of this flag. |
| <i>AVWIN_HASCLOSEBOX</i> | Window has a close box. |
| <i>AVWIN_HASZOOMBOX</i> | Window has a zoom box. UNIX — This flag only adds a Maximize decoration for nonfloating windows under mwm, not for floating or modal windows. No instruction regarding zoom boxes is being set for OpenWindows; the “Full Size” instruction only makes windows full-height, not full-width. |
| <i>AVWIN_HASMINIMIZEBOX</i> | Window has a minimize box. |
| <i>AVWIN_HASDRAGBAR</i> | Window has a drag bar. |
| <i>AVWIN_AUXILIARY</i> | Window is an <i>auxiliary</i> window. Such windows pass unhandled menu commands to the frontmost document window. For example, the Find dialog box is auxiliary; menu items like Save, Close, and so on should still be enabled. A window that does not purport to operate on the topmost document window (for example, an “Establish Connection to Library of Congress” dialog) would probably not be auxiliary. |
| <i>AVWIN_WANTSKEY</i> | The Acrobat viewer may freely assign key status to this window, if appropriate. At any time after the window has been created, this flag may be set or cleared using AVWindowSetWantsKey . |

AVWindowHandler

```
typedef struct _t_AVWindowHandler {
    ASSize_t size;
    AVWindowMouseDownProc MouseDown;
    AVWindowWillCloseProc WillClose;
    AVWindowDidCloseProc DidClose;
    AVWindowDidActivateProc DidActivate;
    AVWindowDidBecomeKeyProc DidBecomeKey;
    AVWindowKeyDownProc KeyDown;
    AVWindowWillResignKeyProc WillResignKey;
    AVWindowWillDeactivateProc WillDeactivate;
    AVWindowDrawProc Draw;
    AVWindowWillBeResizedProc WillBeResized;
    AVWindowPerformEditOpProc PerformEditOp;
    AVWindowCanPerformEditOpProc CanPerformEditOp;
    AVWindowAdjustCursorProc AdjustCursor;
    AVWindowDidResizeProc DidResize;
    /* New in Acrobat 4.0 */
    AVWindowDestroyPlatformThingProc DestroyPlatformThing;
} AVWindowHandlerRec, *AVWindowHandler;
```

| | |
|------------------------|--|
| Description | Data structure containing callbacks that implement a window handler. The callbacks implement the window handler functions. For example, resize the window, draw its contents, handle mouse clicks, handle key presses. <i>NULL</i> values are acceptable; default behavior is then used. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVWindowNew AVWindowNewFromPlatformThing |

size

Size of the data structure. Must be set to *sizeof(AVWindowHandlerRec)*.

AVWindowLayer

| | |
|------------------------|---|
| Description | Constants that specify the layer in which a newly-created <i>AVWindow</i> is to appear. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVWindowNew AVWindowNewFromPlatformThing |

| | |
|------------------------|--|
| <i>AVWLnonfloating</i> | Regular window, such as a document window. |
| <i>AVWLfloating</i> | Floating window, such as a palette. |
| <i>AVWLmodal</i> | Modal dialog. |

AVZoomType

| | |
|------------------------|---|
| Description | Enumerated data type. Specifies the zoom strategy the Acrobat viewer is to follow. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | <u>AVPageViewZoomTo</u> <u>AVPageViewGetZoomType</u> |

| |
|--|
| <i>AVZoomNoVary</i> No variable zoom (that is, <i>zoom</i> is a fixed value such as 1.0 for 100%). |
| <i>AVZoomFitPage</i> Fits the page to the window. |
| <i>AVZoomFitWidth</i> Fits the page width to the window. |
| <i>AVZoomFitHeight</i> Fits the page height to the window. |
| <i>AVZoomFitVisibleWidth</i> Fits the width of the portion of the page upon which marks are made to the window. |
| <i>AVZoomPreferred</i> Uses the page's preferred zoom. |

Character Type Codes

| | |
|------------------------|--|
| Description | Constants that specify a character's type (uppercase, lowercase, number, punctuation, and so forth). |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDWordGetCharacterTypes PDWordSplitString PDWordFilterString |

| Character type | Description |
|----------------------|--|
| <i>W_CNTL</i> | A control code. |
| <i>W_LETTER</i> | A lowercase letter. |
| <i>W_UPPERCASE</i> | An uppercase letter. |
| <i>W_DIGIT</i> | A digit. |
| <i>W_PUNCTUATION</i> | A punctuation mark. |
| <i>W_HYPHEN</i> | A hyphen. |
| <i>W_SOFT_HYPHEN</i> | A hyphen that is only present because a word is broken across two lines of text. |
| <i>W_LIGATURE</i> | A ligature. |
| <i>W_WHITE</i> | A whitespace glyph. |
| <i>W_COMMA</i> | A comma (commas and periods are treated separately from other punctuation marks because they are used both as word punctuation marks and as delimiters in numbers, and need to be treated differently in the two cases). |
| <i>W_PERIOD</i> | A period. |
| <i>W_ACCENT</i> | An accent mark. |
| <i>W_UNMAPPED</i> | A glyph that cannot be represented in the destination font encoding. |
| <i>W_END_PHRASE</i> | An end-of-phrase glyph (for example, ".", "?", "!", ":", and ";"). |
| <i>W_WILD_CARD</i> | A wildcard glyph (for example, "*" and "?") that should not be treated as a normal punctuation mark. |

| Character type | Description |
|---------------------|---|
| <i>W_WORD_BREAK</i> | A glyph that acts as a delimiter between words (see Glyph Names of Word Separators). |

CosDocCreate Flags

| | |
|------------------------|---|
| Description | Flags used to specify the attributes of a CosDoc when it is created by CosDocCreate . |
| Header File | <i>CosExpT.h</i> |
| Related Methods | CosDocCreate |

| | |
|-----------------------------|---|
| <i>cosDocCreateInfoDict</i> | Indicates if an info dictionary should be created for the document. |
|-----------------------------|---|

CosDocOpenParams

```
typedef struct _t_CosDocOpenParams {
    ASSize_t size;
    ASUns32 openFlags;
    ASFileSys fileSys;
    ASPathName pathName;
    const char* headerString;
} CosDocOpenParamsRec, *CosDocOpenParams;
```

| | |
|------------------------|---|
| Description | Parameters used when saving a file using CosDocOpenWithParams . |
| Header File | <i>CosExpT.h</i> |
| Related Methods | CosDocOpenWithParams |

| | |
|---------------------|---|
| <i>size</i> | Size of the data structure. Must be set to <i>sizeof(CosDocOpenParamsRec)</i> . |
| <i>openFlags</i> | Bit field of flags indicating how to open the file. <i>kCosDocOpenDoRepair</i> — Repair the file if necessary. |
| <i>fileSys</i> | The <i>ASFileSys</i> with which the file is opened. May be <i>NULL</i> if using the default file system. |
| <i>pathName</i> | <i>(Required)</i> The <i>ASPathName</i> of the file to open. |
| <i>headerString</i> | Expected header string in file, for example, "%FDF-"; if <i>NULL</i> , assumes "%PDF-" |

CosDocSave Flags

| | |
|------------------------|--|
| Description | Flags used to specify the attributes of a <i>CosDoc</i> when it is saved by CosDocSaveToFile . |
| Header File | <i>CosExpT.h</i> |
| Related Methods | CosDocSaveToFile |

| | |
|------------------------------|---|
| <i>cosSaveGarbageCollect</i> | Indicates whether to delete unreferenced objects before the save. |
| <i>cosSaveFullSave</i> | Indicates whether to write all objects, not just changes. |
| <i>cosSaveBinaryOK</i> | Indicates if binary data can be stored in the file. |

CosDocSaveParams

```
typedef struct _t_CosDocSaveParams {
    ASSize_t size;
    char* header;
    char* cryptData;
    ASInt32 cryptDataLen;
    ProgressMonitor mon;
    void* monClientData; /* client data for progress monitor */
} CosDocSaveParamsRec, *CosDocSaveParams;
```

| | |
|------------------------|--|
| Description | Parameters used when saving a file using CosDocSaveToFile and CosDocSaveWithParams . |
| Header File | <i>CosExpT.h</i> |
| Related Methods | CosDocSaveToFile CosDocSaveWithParams |

| | |
|----------------------|--|
| <i>size</i> | Size of the data structure. Must be set to <i>sizeof(CosDocSaveParamsRec)</i> . |
| <i>header</i> | Complete header string, such as, "%FDF-1.0". |
| <i>cryptData</i> | The encryption key to pass into the PDCryptHandler if security has been set on the document. |
| <i>cryptDataLen</i> | Length of the encryption key, in bytes. Cannot be greater than 5. |
| <i>mon</i> | Progress monitor. Use AVAppGetDocProgressMonitor to obtain the default progress monitor. |
| <i>monClientData</i> | Pointer to user-supplied data to pass to <i>mon</i> each time it is called. |

Cos Object Types

| | |
|------------------------|--|
| Description | Constants that specify a Cos object's type (string, number, dictionary,...). |
| Header File | <i>CosExpT.h</i> |
| Related Methods | CosObjGetGeneration |

| Object type | Description |
|-------------------|---|
| <i>CosNull</i> | A <i>NULL</i> object, or an invalid object. |
| <i>CosInteger</i> | An integer object. |
| <i>CosFixed</i> | A fixed number object. |
| <i>CosBoolean</i> | An <i>ASBool</i> object. |
| <i>CosName</i> | A name object. |
| <i>CosString</i> | A string object. |
| <i>CosDict</i> | A dictionary object. |
| <i>CosArray</i> | An array object. |
| <i>CosStream</i> | A stream object. |

CosStreamOpenMode

| | |
|------------------------|---|
| Description | Enumerated data type. Specifies whether or not filters and decryption should be applied to the stream's data. |
| Header File | <i>CosExpT.h</i> |
| Related Methods | CosStreamOpenStm |

| Mode | Description |
|--------------------------|---|
| <i>cosOpenRaw</i> | The data will be neither filtered nor decrypted. |
| <i>cosOpenUnfiltered</i> | The data will be decrypted but not filtered. |
| <i>cosOpenFiltered</i> | The data will be both decrypted and filtered. (This is the usual case.) |

Emit Flags

| | |
|------------------------|--|
| Description | Enumerated data types used to specify additional print emit options. |
| Header File | <i>AVExpT.h</i> |
| Related Types | AVDocPrintParams |
| Related Methods | AVDocPrintPagesWithParams |

| | |
|-----------------------------------|--|
| <i>kAVEmitHalftones</i> | Emit the halftones specified in the document. |
| <i>kAVEmitSeparableImagesOnly</i> | Raise an error on images that cannot be represented in EPS files, following the separation conventions in Technical Note #5044, <i>Color Separation Conventions for PostScript Language Programs</i> . |
| <i>kAVSuppressCropClip</i> | Do not emit the cropbox page clip. |
| <i>kAVSuppressTransfer</i> | Do not emit the transfer functions in the document. |
| <i>kAVSuppressBG</i> | Do not emit the BlackGeneration in the document. |
| <i>kAVSuppressUCR</i> | Do not emit the UnderColorRemovals in the document. |
| <i>kAVSuppressCJKFontSubst</i> | If the field is set, calls to AVDocPrintPagesWithParams generate PostScript that suppresses printer-based font substitution for CJK fonts. |

EmitFontOptions

| | |
|------------------------|--|
| Description | Enumerated data types used to specify options for printing a file to PostScript. |
| Header File | <i>AVExpT.h</i> |
| Related Types | AVDocPrintParams |
| Related Methods | AVDocPrintPagesWithParams |

| | |
|---------------------------------|--|
| <i>kAVEmitFontNoFonts</i> | <i>(Obsolete after Acrobat 3.0)</i> Embed no fonts. |
| <i>kAVEmitFontAllEmbType1</i> | <i>(Obsolete after Acrobat 3.0)</i> Embed all Type 1 embedded fonts. |
| <i>kAVEmitFontAllType1</i> | <i>(Obsolete after Acrobat 3.0)</i> Embed all Type 1 fonts. |
| <i>kAVEmitFontEmbeddedFonts</i> | Emit all embedded fonts. |
| <i>kAVEmitFontAllFonts</i> | Emit all fonts. |

ExternalDocServerCreationData

```
typedef struct _t_ExternalDocServerCreationData {
    ASSize_t size;
    ExternalDocWindowData platformWindow;
    AVExecuteProc acrobatProc;
    void* acrobatProcData;
    CrossDocLinkProc crossDocLinkProc;
    void* crossDocLinkProcData;
    AVSetMessageProc setMessage;
    void* setMessageProcData;
    /* New for Acrobat 3.01 */
    CrossDocLinkWithDestProc crossDocLinkWithDestProc;
    void* crossDocLinkWithDestData;
} ExternalDocServerCreationDataRec,
*ExternalDocServerCreationData;
```

| | |
|------------------------|--|
| Description | Data for an AVDoc in an external window. Platform-dependent structure used in AVDocOpenParams when opening an AVDoc with AVDocIsReadOnly , AVDocOpenFromFileWithParams , or AVDocOpenFromPDDocWithParams . |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVDocIsReadOnly AVDocOpenFromFileWithParams AVDocOpenFromPDDocWithParams |

| | |
|-----------------------|--|
| <i>size</i> | Size of the data structure. Must be set to <code>sizeof(ExternalDocServerCreationDataRec)</code> |
| <i>platformWindow</i> | Platform-dependent structure of type ExternalDocWindowData representing a window. <ul style="list-style-type: none"> Macintosh — ExternalDocWindowData structure Windows — <i>HWND</i> cast as ExternalDocWindowData UNIX — <i>Widget</i> cast as ExternalDocWindowData |
| <i>acrobatProc</i> | Optional callback. Called when the “Acrobat button” (if present) is clicked in the external application. |

| |
|--|
| <i>acrobatProcData</i> Client-specified data for <i>acrobatProc</i> . |
| <i>crossDocLinkProc</i> Callback to call when a cross-document link occurs. |
| <i>crossDocLinkProcData</i> Client-specified data for <i>crossDocLinkProc</i> . |
| <i>setMessage</i> Currently unused. |
| <i>setMessageProcData</i> Currently unused. Client-specified data for <i>setMessage</i> . |
| <i>crossDocLinkWithDestProc</i> Callback to call when a cross-document link occurs. |
| <i>crossDocLinkWithDestData</i> Client-specified data for <i>crossDocLinkWithDestProc</i> . |

ExternalDocWindowData

```
typedef struct _t_ExternalDocWindowData {
    ExternalDocWindowRefData ref;
    AVSetCursorProc setCursor;
    void* setCursorProcData;
} ExternalDocWindowDataRec, *ExternalDocWindowData;
```

| | |
|------------------------|--|
| Description | <p>Data for an AVDoc in an external window. Platform-dependent structure used in ExternalDocServerCreationData when opening an AVDoc with AVDocIsReadOnly, AVDocOpenFromFileWithParams, or AVDocOpenFromPDDocWithParams.</p> <p>In Mac OS, a plug-in must handle events that effect the window, such as resize and mouse events.</p> |
| Header File | AVExpT.h |
| Related Methods | AVDocIsReadOnly AVDocOpenFromFileWithParams AVDocOpenFromPDDocWithParams |

| | |
|--------------------------|---|
| <i>ref</i> | <p>Pointer to external window data. Must be of type ExternalDocWindowRefData.</p> |
| <i>setCursor</i> | <p>Callback for handling mouse-related events, such as mouse down or mouse movement.</p> |
| <i>setCursorProcData</i> | <p>Optional client-specified data for <i>setCursor</i>.</p> |

ExternalDocWindowRefData

```
typedef struct _t_ExternalDocWindowRefData {
    /* All of these values are handled through dereferences */
    WindowPtr* window;
    ASUns32* x;
    ASUns32* y;
    ASUns32* width;
    ASUns32* height;
    Rect* clip;
    ASInt32* portx;
    ASInt32* porty;
} ExternalDocWindowRefDataRec, *ExternalDocWindowRefData;
```

Description *(Macintosh only)* Data for an external window. Platform-dependent structure used in [ExternalDocWindowData](#) when opening an [AVDoc](#) with [AVDocIsReadOnly](#), [AVDocOpenFromFileWithParams](#), or [AVDocOpenFromPDDocWithParams](#).

Coordinates specified in this structure are in application space. Use [AVRectToRect](#) to translate from user space to device space coordinates, then use the Macintosh *GlobalToLocal* function to translate from device space coordinates to application space coordinates.

Header File *AVExpT.h*

Related Methods [AVDocIsReadOnly](#)
[AVDocOpenFromFileWithParams](#)
[AVDocOpenFromPDDocWithParams](#)

| | |
|---------------|--|
| <i>window</i> | <i>WindowPtr</i> for the external window. |
| <i>x</i> | x-displacement in application space coordinates from the application's window to the AVPageView in which Acrobat renders the PDF file. |
| <i>y</i> | y-displacement in application space coordinates from the application's window to the AVPageView in which Acrobat renders the PDF file. |
| <i>width</i> | Width of external window, specified in device space units. |

height

Height of external window, specified in device space units.

clip

Clipping rectangle (Macintosh *Rect*) for external window in application space units. Usually the entire window.

portx

x-displacement in application space coordinates from the [AVPageView](#) in which Acrobat renders the PDF file to the actual PDF file page. Should usually be 0.

porty

y-displacement in application space coordinates from the [AVPageView](#) in which Acrobat renders the PDF file to the actual PDF file page. Should usually be 0.

Font Flags

| | |
|------------------------|--|
| Description | Constants that indicate a font's attributes (fixed width, roman or symbolic, sans serif, and so forth). |
| Header File | <i>PDExpT.h</i> |
| Related Methods | Part of the PDFontMetricsP structure used by PDFontGetMetrics and PDFontSetMetrics . |

| |
|---|
| <i>PD_FIXED_WIDTH</i> All glyphs in the font are the same width. |
| <i>PD_SERIF</i> The font is a serif font. |
| <i>PD_PI</i> The font is a symbolic (pi) font. |
| <i>PD_SCRIPT</i> The font is a script font. |
| <i>PD_STD_ENCODING</i> The font uses standard encoding. |
| <i>PD_ITALIC</i> The font is an italic font. |
| <i>PD_ALL_CAP</i> The font is an all-caps font. |
| <i>PD_SMALL_CAP</i> The font is a small caps font. |
| <i>PD_FORCE_BOLD</i> Force bold characters to draw bold even at small point sizes. |

gExtensionID

| | |
|------------------------|--|
| Description | A global variable containing an unique identifier for the plug-in. |
| Header File | <i>PICommon.h</i> |
| Related Methods | Any method needing to identify the plug-in. |

gResFile

| | |
|------------------------|--|
| Description | <i>(Macintosh only)</i> A global variable containing the file reference number for the plug-in's file. This is needed with version 2.1 or later of the Acrobat viewers because the plug-in's resource file is not placed at the top of the resource chain automatically. See the Macintosh chapter of Technical Note #5167, Acrobat API Development , for further information. |
| Header File | <i>PICommon.h</i> |
| Related Methods | Any method needing to identify the plug-in. |
| Example | <pre>short oldResFile; DialogPtr myDialog; oldResFile = CurResFile(); UseResFile(gResFile); myDialog = GetNewDialog(23,NULL,(Ptr) -1); useResFile(oldResFile);</pre> |

HFTs

| | |
|------------------------|---|
| Description | Global variables for each of the Acrobat viewer's built-in <i>HFTs</i> . These are needed when replacing a method or calling a replaced method. |
| Header File | PIMain.h |
| Related Methods | HFTReplaceEntry HFTGetReplacedEntry |

| When replacing a method from the API section ... | <i>HFT</i> must be... |
|--|------------------------|
| AVModel | <i>gAcroViewHFT</i> |
| PDModel | <i>gPDModelHFT</i> |
| AcroSupport | <i>gAcroSupportHFT</i> |
| Cos | <i>gCosHFT</i> |
| Macintosh-specific | <i>gMacintoshHFT</i> |
| UNIX-specific | <i>gUnixHFT</i> |
| Windows-specific | <i>gWinHFT</i> |

HFTEntry

```
typedef void* HFTEntry;
```

| | |
|------------------------|---|
| Description | <p>A single entry in an <i>HFT</i>.</p> <p>An <i>HFTEntry</i> may be cast to a pointer to a function whose prototype must be provided by the <i>HFT</i>'s description file.</p> |
| Header File | <i>CoreExpT.h</i> |
| Related Methods | <u>HFTReplaceEntry</u> <u>HFTGetReplacedEntry</u> |

HFTEntryReplaceable

| | |
|------------------------|--|
| Description | <p>A flag that specifies whether or not an <i>HFT</i> entry is replaceable.</p> <p>If set, the new entry can be replaced. Plug-ins should generally use this value, allowing other plug-ins to subsequently replace the method again.</p> <p>If not set, the new entry cannot be replaced.</p> |
| Header File | <i>CoreExpT.h</i> |
| Related Methods | <u>HFTReplaceEntry</u> |

HiliteEntry

```
typedef struct _t_HiliteEntry {  
    ASUnsl6 offset;  
    ASUnsl6 length;  
} HiliteEntry;
```

| | |
|------------------------|--|
| Description | Data structure representing a single entry (starting location and length) in a highlight list. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDTextSelectCreatePageHilite PDTextSelectCreateWordHilite |

MDFFile

```
typedef void* MDFFile;
```

Description

An opaque representation of a file. Different file systems may have different representations for *MDFFiles*. File system implementors may choose any convenient representation for an *MDFFile*. In particular, file systems need not worry about *MDFFile*-space conflicts. The *ASFile* object is guaranteed to be unique across all open files and calls to *ASFile* methods are automatically mapped into calls to *ASFileSystem* callbacks with the corresponding *MDFFile*.

NULL is a legitimate value for *MDFFile*.

Header File

ASExpT.h

Related Methods

[ASFileFromMDFFile](#)
[ASFileGetMDFFile](#)

Modifier Keys

| | |
|------------------------|---|
| Description | Constants corresponding to various keyboard modifier keys. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVSysGetModifiers AVMenuItemGetShortcut AVMenuItemNew |

| | | |
|-------------------|-------------|-----------|
| <i>AV_COMMAND</i> | Command key | — |
| <i>AV_OPTION</i> | Option key | Ctrl key |
| <i>AV_CONTROL</i> | Control key | Ctrl key |
| <i>AV_SHIFT</i> | Shift key | Shift key |

Page Specification

| | |
|------------------------|---|
| Description | Enumerated data type. Can be used wherever a page number or range or count is required. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | Various <i>PDDoc</i> methods. |

| Flag | Description |
|--------------------------|-----------------------------|
| <i>PDBeforeFirstPage</i> | Page before the first page. |
| <i>PDLastPage</i> | Last page. |
| <i>PDAIIPages</i> | All pages of a document. |
| <i>PDOddPagesOnly</i> | Odd pages of a document. |
| <i>PDEvenPagesOnly</i> | Even pages of a document. |

PDAnnot Flags

Description

Constants that indicate annotation properties.

Note: These flags are not the ones used in the flags field of the [AVAnnotHandler](#) structure.

Header File

PDExpT.h

Related Methods

[PDAnnotGetFlags](#)
[PDAnnotSetFlags](#)

pdAnnotInvisible

Controls whether the annotation is invisible—or is drawn with the missing handler icon—if its handler is not available.

pdAnnotHidden

Controls whether the annotation should be drawn—whether its handler is present or not.

pdAnnotPrint

Controls whether the annotation is printed or not.

pdAnnotNoZoom

If set, the annotation does not zoom with the page view.

pdAnnotNoRotate

If set, the annotation does not rotate with the page.

pdAnnotNoView

If set, the annotation is not viewed but can be printed.

pdAnnotReadOnly

If set, the annotation does not interact with the user.

PDAnnotHandler

```
typedef struct _t_PDAnnotHandler {
    ASSize_t size;
    void* userData;
    PDAnnotHandlerGetTypeProc GetType;
    PDAnnotHandlerGetAnnotInfoProc GetAnnotInfo;
    PDAnnotHandlerDeleteAnnotInfoProc DeleteAnnotInfo;
    PDDocWillExportAnnotProc WillExportAnnot;
    PDDocWillImportAnnotProc WillImportAnnot;
    PDAnnotWillPrintProc WillPrintAnnot;
    PDAnnotHandlerGetAnnotInfoFlagsProc GetAnnotInfoFlags;
} PDAnnotHandlerRec, *PDAnnotHandler;
```

Description Data structure containing callbacks that implement an annotation manager. The callbacks implement the annotation manager functions. For example, view, delete, or export the annotations of a document as a list, sorted by type, author, or date.

Header File *PDExpt.h*

Related Callbacks None

Related Methods [PDRegisterAnnotHandler](#)

| | |
|-----------------|---|
| <i>size</i> | Size of the data structure. Must be set to <i>sizeof(PDAnnotHandlerRec)</i> . |
| <i>userData</i> | Pointer to a block of user-supplied data. |

PDAnnotInfo

```
typedef struct _t_PDAnnotInfoRec {
    ASSize_t size;
    ASUns32 nOperationFlags;
    unsigned char* cAuthor;
    unsigned char* cContents;
    unsigned char* cDate;
} PDAnnotInfoRec, *PDAnnotInfo;
```

Description Information associated with an annotation.

Header File *PDExpt.h*

Related Callbacks [PDAnnotHandlerDeleteAnnotInfoProc](#)
[PDAnnotHandlerGetAnnotInfoProc](#)

Related Methods [PDRegisterAnnotHandler](#)

| |
|--|
| <i>size</i> |
| Size of the data structure. Must be set to <i>sizeof(PDAnnotInfoRec)</i> . |
| <i>nOperationFlags</i> |
| Operations allowed on this annotation. Operations permitted are: |
| <ul style="list-style-type: none"> • <i>PDAnnotOperationSummarize</i> — OK to summarize annotations • <i>PDAnnotOperationFilter</i> — OK to filter annotations • <i>PDAnnotOperationManager</i> — OK to manage annotations • <i>PDAnnotIgnorePerms</i> — Allow adding this annotation type to a write-protected document • <i>PDAnnotOperationSummarize</i> — All operations are allowed. |
| <i>cAuthor</i> |
| Author of this annotation. Must be in either PDFDocEncoding or Unicode (with the 0xFEFF prefix). |
| <i>cContents</i> |
| Associated text for this annotation. Must be in either PDFDocEncoding or Unicode (with the 0xFEFF prefix). |
| <i>cDate</i> |
| Modification date of this annotation. The date should be the standard date format as specified in Section 7.2, "Date", in the Portable Document Format Reference Manual . |

PDCharSet

| | |
|------------------------|---|
| Description | Enumerated data type. Identifies the character set of a Type 1, multiple master Type 1, or TrueType font. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDFontGetCharSet |

| |
|---|
| <i>PDStandardRomanCharSet</i> |
| The font uses Adobe Standard encoding. This is determined by the "Uses Adobe Standard Encoding" bit in the font descriptor. |
| <i>PDUnknownCharSet</i> |
| The font does not use Adobe Standard Encoding. |
| <i>PDAdobeExpertCharSet</i> |
| Currently unused. |

PDChromaticity

```
typedef struct _t_PDChromaticity {  
    ASFixed x;  
    ASFixed y;  
} PDChromaticity;
```

| | |
|------------------------|---|
| Description | Data structure containing a monitor's chromaticity, for use in displaying device-independent color. x and y are the two values needed to specify the chromaticity. |
| Header File | <i>PDExpT.h</i> |
| Related Types | PDColorCalP |
| Related Methods | PDPrefGetColorCal PDPrefSetColorCal |

PDColorCal

PDColorCalP

```
typedef struct _t_PDColorCal {  
    PDChromaticity whiteChrom;  
    PDChromaticity redChrom;  
    PDChromaticity greenChrom;  
    PDChromaticity blueChrom;  
    ASFixed redGamma;  
    ASFixed greenGamma;  
    ASFixed blueGamma;  
} PDColorCal, *PDColorCalP;
```

| | |
|------------------------|--|
| Description | Data structure used to represent the characteristics of an output device; needed for device-independent color. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPrefGetColorCal PDPrefSetColorCal |

PDColorSpace

| | |
|------------------------|---|
| Description | Enumerated data type. Specifies the color space in which a color value is specified (for example, RGB or grayscale). |
| Header File | <i>PDExpT.h</i> |
| Related Types | PDColorValue |
| Related Methods | AVAppBeginFullScreen AVPageViewGetColor AVPageViewSetColor PDAnnotGetColor PDAnnotSetColor PDStyleGetColor |

| |
|--|
| <i>PDDeviceGray</i> |
| Grayscale. Requires 1 <i>value</i> entry to specify the color. |
| <i>PDDeviceRGB</i> |
| Red–Green–Blue color specification. Requires 3 <i>value</i> entries to specify the color. |
| <i>PDDeviceCMYK</i> |
| Cyan–Magenta–Yellow–Black color specification. Requires 4 <i>value</i> entries to specify the color. |

PDColorValue

```
typedef struct _t_PDColorValueRec {  
    PDColorSpace space;  
    ASFixed value[4];  
} PDColorValueRec, *PDColorValue;
```

Description Data structure representing a color. The number of elements needed in the *value* field depends on the color space type (specified in the *space* field). See [PDColorSpace](#) for more information.

See also [AVPrefsType](#).

Header File *PDExpT.h*

Related Methods [AVAppBeginFullScreen](#)
[AVPageViewGetColor](#)
[AVPageViewSetColor](#)
[PDAnnotGetColor](#)
[PDAnnotSetColor](#)
[PDStyleGetColor](#)

PDFCryptHandler

```
typedef struct _t_PDFCryptHandler {
    ASSize_t size;
    PDFCryptAuthorizeProc Authorize;
    PDFCryptNewAuthDataProc NewAuthData;
    PDFCryptGetAuthDataProc GetAuthData;
    PDFCryptNewSecurityDataProc NewSecurityData;
    PDFCryptValidateSecurityDataProc ValidateSecurityData;
    PDFCryptUpdateSecurityDataProc UpdateSecurityData;
    PDFCryptNewCryptDataProc NewCryptData;
    PDFCryptFillEncryptDictProc FillEncryptDict;
    PDFCryptGetSecurityInfoProc GetSecurityInfo;
    /* New calls for Acrobat 3.0 */
    PDFCryptFreeSecurityDataProc FreeSecurityData;
    PDFCryptFreeAuthDataProc FreeAuthData;
    PDFCryptFreeCryptDataProc FreeCryptData;
} PDFCryptHandlerRec, *PDFCryptHandler;
```

Description Data structure containing callbacks that implement a security handler. The callbacks implement the security handler functions. For example, get authorization data such as a password from the user, set permissions, read and write security-related data in a PDF file, and so on.

Header File *PDFExpt.h*

Related Methods [PDFRegisterCryptHandler](#)

size

Size of the data structure. Must be set to *sizeof(PDFCryptHandlerRec)*.

PDDocFlags

| | |
|------------------------|---|
| Description | Enumerated data type. Specifies various file status attributes. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDDocClearFlags PDDocGetFlags PDDocSetFlags |

| Flag | | Description |
|------------------------------|----------|---|
| <i>PDDocNeedsSave</i> | get/set | Document has been modified and needs to be saved. |
| <i>PDDocRequiresFullSave</i> | set only | Document cannot be saved incrementally; when it is saved using PDDocSave , the <i>PDSaveFull</i> flag must be specified (see PDSaveFlags). |
| <i>PDDocIsModified</i> | get only | Document has been modified slightly (such as bookmarks or text annotations have been opened or closed), but not in a way that warrants saving. |
| <i>PDDocDeleteOnClose</i> | get/set | Document is based on a temporary file that must be deleted when the document is closed or saved. |
| <i>PDDocWasRepaired</i> | get only | Document was repaired when it was opened. |
| <i>PDDocNewMajorVersion</i> | get only | Document's major version is newer than current. |
| <i>PDDocNewMinorVersion</i> | get only | Document's minor version is newer than current. |
| <i>PDDocOldVersion</i> | get only | Document's version is older than current. |
| <i>PDDocSuppressErrors</i> | get/set | Don't display errors. |
| <i>PDDocIsEmbedded</i> | get/set | Document is embedded in a compound document (OLE, OpenDoc). |
| <i>PDDocIsLinearized</i> | get only | Document is linearized for page-served remote (network) access. |

| Flag | | Description |
|-------------------------|----------|--|
| <i>PDDocIsOptimized</i> | set only | Document is optimized. If this flag is cleared, the Batch Optimizer plug-in and the Adobe PDF Library do not save the file optimized. You can, therefore, linearize a PDF file without optimizing it. Optimizing without linearizing is <i>not</i> allowed, however. |

PDDocInsertPagesParams

```
typedef struct _t_PDDocInsertPagesParams {
    ASSize_t size;
    PDDoc targetDoc;
    ASInt32 insertAfterThisPage;
    PDDoc srcDoc;
    ASInt32 srcFromPage;
    ASInt32 srcToPage;
    ASUns16 insertFlags;
    ASInt32 error;
} PDDocInsertPagesParamsRec, *PDDocInsertPagesParams;
```

| | |
|--------------------------|---|
| Description | Structure used to pass information to PDDocWillInsertPagesEx and PDDocDidInsertPagesEx notifications. |
| Header File | <i>PDExpt.h</i> |
| Related Callbacks | None |
| Related Methods | PDDocInsertPages |

| | |
|----------------------------|--|
| <i>size</i> | Size of the data structure. Must be set to <code>sizeof(PDDocInsertPagesParamsRec)</code> . |
| <i>targetDoc</i> | The document into which pages are inserted. This document must have at least one page. |
| <i>insertAfterThisPage</i> | The page number in <i>targetDoc</i> after which pages from <i>srcDoc</i> are inserted. The first page is 0. If <i>PDBeforeFirstPage</i> (see <i>PDExpT.h</i>) is used, the pages are inserted before the first page in <i>targetDoc</i> . Use <i>PDLastPage</i> to insert pages after the last page in <i>targetDoc</i> . |
| <i>srcDoc</i> | The document containing the pages that are inserted into <i>targetDoc</i> . |
| <i>srcFromPage</i> | The page number of the first page in <i>srcDoc</i> to insert into <i>targetDoc</i> . The first page is 0. |
| <i>srcToPage</i> | The page number of the last page in <i>srcDoc</i> to insert into <i>targetDoc</i> . |

insertFlags

Flags that determine what additional information is copied from *srcDoc* into *targetDoc*. Must be an OR of the following (see *PDExpT.h*):

PDInsertAll — Inserts the entire document *srcDoc* into the document *targetDoc*. This operation not only inserts the pages themselves, but also merges other document data from *srcDoc* into *targetDoc*. In particular, the following happens:

1. The bookmark tree of *srcDoc* is merged into the bookmark tree of *targetDoc* by copying it as a new first-level subtree of *targetDoc*'s bookmark tree root, of which it becomes the last child. If *targetDoc* has no bookmark tree, it acquires one identical to the bookmark tree from *srcDoc*.
2. Named destinations from *srcDoc* (of PDF 1.1 and later) are copied into *targetDoc*. If there are named destinations in *srcDoc* with the same name as some named destination in *targetDoc*, the ones in *targetDoc* retain their names and the copied named destinations are given new names based on the old ones with distinguishing digits added. Actions and bookmarks referring to the old names are made to refer to the new names after being copied into *targetDoc*.
3. Document logical structure from *srcDoc* is copied into *targetDoc*. The top-level children of the structure tree root of *srcDoc* are copied as new top-level children of the structure tree root of *targetDoc*; a structure tree root is created in *targetDoc* if there was none before. The role maps of the two structure trees are merged, with name conflicts resolved in favor of the role mappings present in *targetDoc*. Attribute objects are not copied, nor is class map information from *srcDoc* merged into that for *targetDoc*.

If *PDInsertAll* flag is *not* set, pages copied from *srcDoc* into *targetDoc* will have their structure backpointer information stripped off.

PDInsertBookmarks — Inserts bookmarks as well as pages.

PDInsertThreads — Inserts threads as well as pages.

error

Error code, which is only valid for the [PDDocDidInsertPagesEx](#) notification.

PDDocOpenParams

```
typedef struct _t_PDDocOpenParams{
    ASSize_t size;
    ASFile file;
    ASPathName fileName;
    ASFileSys fileSys;
    PDAuthProcEx authProcEx;
    void* authProcClientData;
    ASBool doRepair;
    PDPerms restrictPerms;
} PDDocOpenParamsRec, *PDDocOpenParams;
```

Description Structure used by [PDDocOpenWithParams](#) to specify file open information. The parameters are very similar to those in [PDDocOpenEx](#) and [PDDocOpenFromASFileEx](#).

Header File *PDExpt.h*

Related Callbacks None

Related Methods [PDDocOpenWithParams](#)

| | |
|-----------------|--|
| <i>size</i> | Size of the data structure. Must be set to <i>sizeof(PDDocOpenParamsRec)</i> . |
| <i>file</i> | Path name to the file, specified in whatever format is correct for <i>fileSys</i> . |
| <i>fileName</i> | Path name to the file, specified in whatever format is correct for <i>fileSys</i> . |
| <i>fileSys</i> | Pointer to an ASFileSysRec containing the file system in which the file resides. |

authProcEx

Authorization callback, called only if the file is encrypted. This callback should obtain whatever information is needed to determine whether or not the user is authorized to open the file, then call *PDDocAuthorize* (which returns the permissions that the authentication data enables). If the file is encrypted and *authProcEx* is *NULL* or returns *false*, *pdErrPassword* is raised.

The Acrobat viewer's built-in authorization procedure requires the user to enter a password, and allows the user to try three times before giving up.

authProcClientData

Pointer to user-specified data to pass to *authProcEx* each time it is called.

doRepair

If *true*, attempt to repair the file if it is damaged; if *false*, do not attempt to repair the file if it is damaged.

restrictPerms

The permissions to remove from the document.

PDDocPreSaveInfo

```
typedef struct _t_PDDocPreSaveInfo {  
    ASSize_t size;  
    CosObjSetCallbackFlagProc callbackProc;  
} PDDocPreSaveInfoRec, *PDDocPreSaveInfo;
```

Description Structure used to flag Cos objects you wish to access while a *PDDoc* is being saved.

Header File *PDExpt.h*

Related Callbacks [CosObjSetCallbackFlagProc](#)

Related Methods [PDDocOpenWithParams](#)

| |
|---|
| <i>size</i> |
| Size of the data structure. Must be set to <i>sizeof(PDDocPreSaveInfo)</i> . |
| <i>callbackProc</i> |
| CosObjSetCallbackFlagProc callback to set flags in Cos objects to provide access to them during the save. |

PDDocReadAhead Flags

| | |
|------------------------|---|
| Description | Flags describing what type of data to read "ahead." |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDDocReadAhead |

kPDDocReadAheadAcroForms

Allows the AcroForm plug-in to request that all the AcroForm data be read "ahead," before the viewer needs it. This flag is ignored if the PDF file does not contain a Forms hint table. See Section 9.4 in the [Portable Document Format Reference Manual](#) for information about the Forms hint table.

kPDDocReadAheadTemplates

Requests that the PDF file's Forms Template data be read "ahead," before the viewer needs it. There is currently no Template hint table defined, so this flag simply causes the rest of the file to be read.

kPDDocReadAheadPageLabels

Requests that the PDF file's page label data be read "ahead," before the viewer needs it. There is currently no page label hint table defined, so this flag simply causes the rest of the file to be read.

kPDDocReadAheadStructure

Requests that the PDF file's logical structure data be read "ahead," before the viewer needs it. There is currently no logical structure hint table defined, so this flag simply causes the rest of the file to be read.

PDDocSaveParams

```
typedef struct _t_PDDocSaveParams {
    ASSize_t size;
    PDSaveFlags saveFlags;
    ASPathName newPath;
    ASFileSys fileSys;
    ProgressMonitor mon;
    void* monClientData;
    CancelProc cancelProc;
    void* cancelProcClientData;
    /* Added in Acrobat 4.0 */
    PDDocPreSaveProc preSaveProc;
    void* preSaveProcClientData;
    CosObjOffsetProc offsetProc;
    void* offsetProcClientData;
    ASInt16 major;
    ASInt16 minor;
} PDDocSaveParamsRec, *PDDocSaveParams;
```

Description Parameters used when saving a file with [PDDocSaveWithParams](#) and returned by the [PDDocWillSaveEx](#) notification. Most of these parameters are the same as the parameters for [PDDocSave](#).

Header File *PDExpT.h*

Related Methods [PDDocSaveWithParams](#)

| | |
|------------------|--|
| <i>size</i> | Size of the data structure. Must be set to <i>sizeof(PDDocSaveParamsRec)</i> |
| <i>saveFlags</i> | Must be one of the PDSaveFlags values. |
| <i>newPath</i> | <p>The path to which the file is saved. A path must be specified when either <i>PDSaveFull</i> or <i>PDSaveCopy</i> are used for <i>saveFlags</i>. If <i>PDSaveIncremental</i> is specified in <i>saveFlags</i>, then <i>newPath</i> should be <i>NULL</i>.</p> <p>If <i>PDSaveFull</i> is specified and <i>newPath</i> is the same as the file's original path, the new file is saved to a file system determined temporary path, then the old file is deleted and the new file is renamed to <i>newPath</i>.</p> |

| |
|---|
| <i>fileSys</i> |
| File system. If <i>NULL</i> , uses the <i>fileSys</i> of the document's current backing file. Implementation Restriction: Files can only be saved to the same file system, thus <i>fileSys</i> must be <i>NULL</i> or an error is raised. |
| <i>mon</i> |
| Progress monitor. Use AVAppGetDocProgressMonitor to obtain the default. |
| <i>monClientData</i> |
| Pointer to user-supplied data to pass to <i>mon</i> each time it is called. |
| <i>cancelProc</i> |
| A cancel procedure. Use AVAppGetCancelProc to obtain the current cancel procedure. |
| <i>cancelProcClientData</i> |
| Pointer to user-specified data to pass to <i>cancelProc</i> each time it is called. |
| <i>preSaveProc</i> |
| Callback to flag Cos objects you wish to access while the <i>PDDoc</i> is being saved. |
| <i>preSaveProcClientData</i> |
| Pointer to user-specified data to pass to <i>preSaveProc</i> each time it is called. |
| <i>offsetProc</i> |
| Callback to get information about interesting Cos objects while the <i>PDDoc</i> is being saved. |
| <i>offsetProcClientData</i> |
| Pointer to user-specified data to pass to <i>offsetProc</i> each time it is called. |
| <i>major</i> |
| Major PDF version number of the document. If <i>major</i> equals 0, both <i>major</i> and <i>minor</i> are ignored. Make sure that the document conforms to the version number you are setting. |
| <i>minor</i> |
| Minor PDF version number of the document. |

PDEColorSpaceStruct

```
typedef union {  
    PDEGrayCalFlt* calGray;  
    PDERGBCalFlt* calRGB;  
    PDELabCalFlt* lab;  
    PDEICCBasedColorData* icc;  
    PDEIndexedColorData* indexed;  
    PDEPatternColorSpace patternbase;  
    PDESeparationColorData* sep;  
    PDEDeviceNColorData* devn;  
} PDEColorSpaceStruct;
```

| | |
|--------------------------|--|
| Description | A color space structure for PDEColorSpaceCreate . See Section 7.11 in the Portable Document Format Reference Manual for information on color spaces. |
| Header File | <i>PEExpt.h</i> |
| Related Callbacks | None |
| Related Methods | PDEColorSpaceCreate |

PDEColorSpec

PDEColorSpecP

```
typedef struct _t_PDEColorSpec {  
    PDEColorSpace space;  
    PDEColorValue value;  
} PDEColorSpec, *PDEColorSpecP;
```

Description Structure describing a color specification.

Header File *PEExpT.h*

Related Methods Numerous

space

The specified *PDEColorSpace*.

value

The color value.

PDECOLORVALUE

PDECOLORVALUEP

```
typedef struct _t_PDECOLORVALUE {
    ASFixed color[7];
    PDEObject colorObj2;
    PDEObject colorObj;
} PDECOLORVALUE, *PDECOLORVALUEP;
```

| | |
|------------------------|-------------------------------------|
| Description | Structure describing a color value. |
| Header File | <i>PEExpT.h</i> |
| Related Methods | Numerous |

color

Color value components. For instance, a Gray color space has 1 component, an RGB color space has 3 components, a CMYK has 4 components, and so on.

colorObj2

For a DeviceN color space.

colorObj

For color spaces whose color values do not have numeric values, such as the Pattern and Separation color spaces.

PDEContentAttrs

PDEContentAttrsP

```
typedef struct _t_PDEContentAttrs {
    ASUns32 flags;
    ASFixed cacheDevice[8];
    ASInt32 formType;
    ASFixedRect bbox;
    ASFixedMatrix matrix;
    CosObj XUID;
} PDEContentAttrs, *PDEContentAttrsP;
```

| | |
|------------------------|--|
| Description | Structure describing attributes of a PDEContent object. |
| Header File | <i>PEExpT.h</i> |
| Related Methods | PDEContentGetAttrs PDEContentToCosObj |

| |
|---|
| <i>flags</i> |
| PDEContentFlags flags. |
| <i>cacheDevice</i> |
| If <i>flags</i> has <i>kPDESetCacheDevice</i> set, the first 6 cache device values contain the operands for the d1 (setcachdevice) page operator. If <i>flags</i> has <i>kPDESetCacheDevice</i> set, <i>cacheDevice</i> contains 2 charwidth values. |
| <i>formType</i> |
| Only used if PDEContent contains a Form <i>XObject</i> . Corresponds to FormType key in the <i>XObject</i> Form attributes dictionary. |
| <i>bbox</i> |
| Only used if PDEContent contains a Form. Bounding box of the PDEContent object. Corresponds to BBox key in the <i>XObject</i> Form attributes dictionary. |
| <i>matrix</i> |
| Only used if PDEContent contains a Form. Transformation matrix for the PDEContent object. Corresponds to Matrix key in the <i>XObject</i> Form attributes dictionary. |

XUID

Only used if [PDEContent](#) contains a Form. The form's XUID, an ID that uniquely identifies the form. Corresponds to **XUID** key in the *XObject* Form attributes dictionary.

PDEContentFlags

```
typedef enum {  
    kPDESetCacheDevice = 0x0001,  
    kPDESetCharWidth = 0x0002,  
    kPDEFormMatrix = 0x0004  
} PDEContentFlags;
```

Description Bit field for [PDEContentAttrs](#).

Header File *PEExpT.h*

Related Methods [PDEContentGetAttrs](#)
[PDEContentToCosObj](#)

| |
|--|
| <i>kPDESetCacheDevice</i> |
| Indicates <i>cacheDevice</i> contains 6 cache device values. |
| <i>kPDESetCharWidth</i> |
| Indicates <i>cacheDevice</i> contains 2 charwidth values. |
| <i>kPDEFormMatrix</i> |
| Indicates <i>formMatrix</i> contains a valid matrix. |

PDEContentGetResourceFlags

```
typedef enum {  
    kPDEGetFonts,  
    kPDEGetXObjects,  
    kPDEGetColorSpaces  
} PDEContentGetResourceFlags;
```

Description Bit field for [PDEContentAttrs](#).

Header File *PEExpT.h*

Related Methods [PDEContentGetResources](#)

| |
|----------------------------------|
| <i>kPDEGetFonts</i> |
| Obtain font resources. |
| <i>kPDEGetXObjects</i> |
| Obtain <i>Xobject</i> resources. |
| <i>kPDEGetColorSpaces</i> |
| Obtain color space resources. |

PDEContentToCosObjFlags

```
typedef enum {
    kPDEContentToPage = 0x0001,
    kPDEContentToForm = 0x0002,
    kPDEContentToCharProc = 0x0004,
    kPDEContentRev1Compat = 0x0008,
    kPDEContentDoNotResolveForms = 0x0010,
    kPDEContentDoNotResolveType3 = 0x0020,
    kPDEContentEmitDefaultRGBAndGray = 0x0040
} PDEContentToCosObjFlags;
```

Description Bit field for the [PDEContentToCosObj](#) method, indicating the type of object to create and how it is created.

Header File *PEExpT.h*

Related Methods [PDEContentToCosObj](#)

| |
|--|
| <i>kPDEContentToPage</i> |
| Create page contents. |
| <i>kPDEContentToForm</i> |
| Create a form. |
| <i>kPDEContentToCharProc</i> |
| Create charprocs. |
| <i>kPDEContentRev1Compat</i> |
| Currently unused. |
| <i>kPDEContentDoNotResolveForms</i> |
| Do not enumerate <i>XObject</i> Forms and their resources. Use the <i>kPDEContentDoNotResolveForms</i> flag in <i>PDEContentToCosObjFlags</i> when you call PDEContentToCosObj to get a Cos object from a <i>PDEContent</i> —not when you get a Cos object for an <i>XObject</i> Form itself. |
| <i>kPDEContentDoNotResolveType3</i> |
| Currently unused. |

kPDEContentEmitDefaultRGBAndGray

Emit calibrated RGB and gray information using the PDF 1.0 compatible mechanism. In this case, generate **rg** and **k** page operators and place **DefaultGray** and **DefaultRGB** color space arrays in the Resources dictionary, as described in Section 7.11.12 in the [*Portable Document Format Reference Manual*](#).

PDEDash

PDEDashP

```
typedef struct _t_PDEDash {  
    ASFixed dashPhase;  
    ASInt32 dashLen;  
    ASFixed dashes[11];  
} PDEDash, *PDEDashP;
```

| | |
|------------------------|---|
| Description | Structure describing a dash specification, as described in Section 6.6 in the Portable Document Format Reference Manual . See Section 8.4.3 for more information on line dash patterns. |
| Header File | <i>PEExpT.h</i> |
| Related Methods | Numerous |

dashPhase

Dash phase. Phase is a number that specifies a distance in user space into the dash pattern at which to begin marking the path.

dashLen

Number of entries in the dash array, an element of the Border array.

dashes

Dash array, which specifies distances in user space for the length of dashes and gaps.

PDEDeviceNColorData

```
typedef struct _t_PDEDeviceNColorData {  
    ASSize_t size;  
    ASAtom* names;  
    ASUns32 nNames;  
    PDEColorSpace alt;  
    CosObj tintTransform;  
} PDEDeviceNColorData;
```

| | |
|--------------------------|---|
| Description | Structure describing a DeviceRGB or DeviceCMYK color space. |
| Header File | <i>PEExpt.h</i> |
| Related Types | PDEColorSpaceStruct |
| Related Callbacks | None |
| Related Methods | PDEColorSpaceCreate |

| | |
|----------------------|---|
| <i>size</i> | Size of the data structure. Must be set to <i>sizeof(PDEDeviceNColorData)</i> . |
| <i>names</i> | Names of colorants. |
| <i>nNames</i> | Number of colorants. |
| <i>alt</i> | Alternative colorspace. |
| <i>tintTransform</i> | The tintTransform dictionary or function. See Section 7.11 in the Portable Document Format Reference Manual for more information. |

PDEElementCopyFlags

```
typedef enum {  
    kPDEElementCopyForClip = 0x0001,  
    kPDEElementCopyClipping = 0x0002  
} PDEElementCopyFlags;
```

Description Bit field for [PDEElementCopy](#).

Header File *PEExpT.h*

Related Methods [PDEElementCopy](#)

| |
|--|
| <i>kPDEElementCopyForClip</i> |
| Copied element does not need <i>gstate</i> or clip. |
| <i>kPDEElementCopyClipping</i> |
| Acquire the clip path and put it in the copied object. |

PDEEnumElementsFlags

```
typedef enum {  
    kPDEContentIgnoreMarkedContent = 0x0001  
} PDEEnumElementsFlags;
```

Description Bit field for [PDEEnumElements](#) method.

Header File *PEExpT.h*

Related Methods [PDEEnumElements](#)

kPDEContentIgnoreMarkedContent

Indicates whether Marked Content is ignored in the enumeration. This may be useful when generating elements purely for display purposes.

PDEFilterArray

PDEFilterArrayP

```
typedef struct _t_PDEFilterArray {
    ASInt32 numFilters;
    PDEFilterSpec spec[1];
} PDEFilterArray, *PDEFilterArrayP;
```

Description

Filter information for streams.

Although the [PDEFilterSpec](#) is declared as a one-member array in the header file, more members can be added by dynamically allocating the *PDEFilterArray* with space for as many filters as you would like to add.

In practice, there is seldom need for more than two filters.

Header File

PEExpT.h

Related Methods

[PDEContentToCosObj](#)
[PDEImageCreate](#)

Example

```
pdeFilterArray = (PDEFilterArrayP)
    malloc(offsetof(PDEFilterArray, spec) +
        (sizeof(PDEFilterSpec) * 2));
```

numFilters

Number of filters in the array.

spec

Variable length array of filter spec.

PDEFILTERSpec

PDEFILTERSpecP

```
typedef struct _t_PDEFILTERSpec {
    CosObj decodeParms;
    CosObj encodeParms;
    ASAtom name;
    ASInt16 padding;
} PDEFILTERSpec, *PDEFILTERSpecP;
```

Description Filter element in a filter array.

Header File *PEExpT.h*

Related Methods [PDEContentToCosObj](#)
[PDEImageCreate](#)

Example

```
CosObj dctDict = CosNewDict(theCosDoc,false,
    3);
CosDictPut(dctDict,
    ASAtomFromString("Columns"),
    CosNewInteger(NULL,false,width_of_image));

CosDictPut(dctDict,
    ASAtomFromString("Rows"),
    CosNewInteger(NULL, false,
    hight_of_image));

CosDictPut(dctDict,
    ASAtomFromString("Colors"),
    CosNewInteger(NULL, false,num_colors));
/* 3 for RGB 4 for CMYK */

pdeFilters->spec[i].encodeParms = dctDict;
```

decodeParms

Parameters used by the decoding filters specified with the Filter key. Corresponds to the **DecodeParms** key in the stream dictionary.

Must be set to *NULL* if *PDEFilterSpec* is specified but no encode or decode parameters are specified. This can be done by passing *CosNewNull* for the unused encode and/or decode params.

Required decode params for **DCTDecode** are **Columns**, **Rows**, and **Colors**. The parameters should be passed in a *CosDict*.

encodeParms

Parameters used when encoding the stream. Required for **DCTDecode** filter; optional for other filters.

Must be set to *NULL* if *PDEFilterSpec* is specified but no encode or decode parameters are specified. This can be done by passing *CosNewNull* for the unused encode and/or decode params.

name

Filter name. Supported filters are: **ASCIIHexDecode**, **ASCII85Decode**, **LZWDecode**, **DCTDecode**, **CCITTFaxDecode**, **RunLengthDecode**, and **FlateDecode**.

padding

Reserved — used to align to 32 bits.

PDEFontAttrs

PDEFontAttrsP

```
typedef struct _t_PDEFontAttrs {  
    ASAtom name;  
    ASAtom type;  
    ASAtom charSet;  
    ASAtom encoding;  
    ASUns32 flags;  
    ASFixedRect fontBBox;  
    ASInt16 missingWidth;  
    ASInt16 stemV;  
    ASInt16 stemH;  
    ASInt16 capHeight;  
    ASInt16 xHeight;  
    ASInt16 ascent;  
    ASInt16 descent;  
    ASInt16 leading;  
    ASInt16 maxWidth;  
    ASInt16 avgWidth;  
    ASInt16 italicAngle;  
    /* New in Acrobat 4.0 */  
    ASAtom cidFontType;  
    ASInt16 wMode;  
    ASAtom psName;  
    ASAtom lang;  
    ASAtom registry;  
    ASAtom ordering;  
    ASInt32 supplement;  
} PDEFontAttrs, *PDEFontAttrsP;
```

| | |
|------------------------|---|
| Description | Attributes of a PDEFont and of a PDSysFont . The fields after encoding are almost the same as a PDFontMetricsP structure. |
| Header File | <i>PEExpT.h</i> |
| Related Methods | PDEFontCreate PDEFontCreateWithParams PDFindSysFont PDSysFontAcquirePlatformData |

| | |
|---------------------|---|
| <i>name</i> | An <i>ASAtom</i> for font name, as in "Times-Roman." Corresponds to the BaseFont key in the font dictionary of a PDF file (see Section 7.7 in the Portable Document Format Reference Manual). |
| <i>type</i> | An <i>ASAtom</i> for font type, corresponding to the Subtype key in a font dictionary. May be "Type1," "TrueType," "MMType1," or "Type0." |
| <i>charSet</i> | An <i>ASAtom</i> for "Roman" or <i>ASAtomNull</i> . If "Roman," the characters must be a subset of the Adobe Standard Roman Character Set. |
| <i>encoding</i> | An <i>ASAtom</i> for font encoding, as in WinAnsiEncoding . |
| <i>flags</i> | Desired font flags, one or more of Font Flags . |
| <i>fontBBox</i> | Font bounding box in 1000 EM units. |
| <i>missingWidth</i> | Width of missing character (.notdef). |
| <i>stemV</i> | Vertical stem width. |
| <i>stemH</i> | Horizontal stem width. |
| <i>capHeight</i> | Capital height. |
| <i>xHeight</i> | X height. |
| <i>ascent</i> | Maximum ascender height. |
| <i>descent</i> | Maximum descender depth. |
| <i>leading</i> | Additional leading between lines. |

| | |
|--------------------|--|
| <i>maxWidth</i> | Maximum character width. |
| <i>avgWidth</i> | Average character width. |
| <i>italicAngle</i> | Italic angle in degrees, if any. |
| <i>cidFontType</i> | ASAtom representing the CID font type: CIDFontType0 or CIDFontType2. |
| <i>wMode</i> | Writing mode. Must be one of: <ul style="list-style-type: none">• 0 for horizontal writing• 1 for vertical writing |
| <i>psName</i> | ASAtom representing the PostScript name of a TrueType font. |
| <i>lang</i> | ASAtom representing the ISO 639 language code. These are listed in Appendix I in the Portable Document Format Reference Manual |
| <i>registry</i> | ASAtom representing the CIDFont's Registry information, as in "Adobe-Japan". |
| <i>ordering</i> | ASAtom representing the CIDFont's Ordering information, for example, "1". |
| <i>supplement</i> | The SystemSupplement field from the CIDFont. |

PDFontCreateFlags

```
typedef enum {
    kPDFontCreateEmbedded,
    kPDFontWillSubset,
    kPDFontDoNotEmbed,
    kPDFontEncodeByGID
} PDFontCreateFlags;
```

Description Flags for [PDFontCreateFromSysFont](#).

Header File *PEExpT.h*

Related Methods *kPDFontWillSubset*

Subset the font. If you want to subset a font, set both the *kPDFontCreateEmbedded* and *kPDFontWillSubset* flags.

kPDFontCreateEmbedded

Create an embedded font.

kPDFontWillSubset

Subset the font. If you want to subset a font, set both the *kPDFontCreateEmbedded* and *kPDFontWillSubset* flags.

kPDFontDoNotEmbed

Do not embed the font.

kPDFontEncodeByGID

Create a CIDFont with identity (GID) encoding.

PDFontCreateParams

PDFontCreateParamsP

```
typedef struct _t_PDFontCreateParams {
    PDFontAttrsP attrsP;
    ASUns32 attrsSize;
    ASInt32 firstChar;
    ASInt32 lastChar;
    ASInt16* widthsP;
    char** encoding;
    ASAtom encodingBaseName;
    ASStm fontStm;
    ASInt32 len1;
    ASInt32 len2;
    ASInt32 len3;
    ASBool hasDW;
    ASInt32 dw;
    CosObj w;
    ASBool hasDW2;
    ASInt32 dw2[2];
    CosObj w2;
    ASInt32 toUnicodeLen;
    ASStm toUnicodeStm;
    ASStm cidToGidMapStm;
    char* panoseNo;
    CosObj fd;
    ASStm cidSetStm;
    ASUns32 flags;
    /* New in Acrobat 4.0 */
    ASFixed* mmDesignVec;
} PDFontCreateParamsRec, *PDFontCreateParamsP;
```

| | |
|--------------------------|--|
| Description | Parameters used for PDFontCreateWithParams to describe a font. |
| Header File | <i>PEExpt.h</i> |
| Related Types | PDEColorSpaceStruct |
| Related Callbacks | None |
| Related Methods | PDFontCreate PDFontCreateWithParams |

| | |
|-------------------------|--|
| <i>attrsP</i> | Pointer to a <i>PDEFontAttrs</i> for the font attributes. |
| <i>attrsSize</i> | Size of the data structure. Must be set to <i>sizeof(PDEFontAttrs)</i> . |
| <i>firstChar</i> | First character index for the widths array, <i>widthsP</i> . |
| <i>lastChar</i> | Last character index for the widths array, <i>widthsP</i> . |
| <i>widthsP</i> | Pointer to widths array. |
| <i>encoding</i> | An array of 256 pointers to glyph names specifying the custom encoding. If any pointer is <i>NULL</i> , no encoding information is written for that entry. |
| <i>encodingBaseName</i> | An <i>ASAtom</i> representing the encoding base name if the encoding is a custom encoding. If <i>encoding</i> is <i>NULL</i> , <i>encodingBaseName</i> is used as the value of the encoding and must be one of "WinAnsiEncoding", "MacRomanEncoding", or "MacExpertEncoding". If no encoding value is desired, use <i>ASAtomNull</i> . |
| <i>fontStm</i> | Stream with font information. |
| <i>len1</i> | Length in bytes of the ASCII portion of the Type 1 font file after it has been decoded. For other font formats, such as TrueType or CFF, only <i>len1</i> is used, and it is the size of the font. |
| <i>len2</i> | Length in bytes of the encrypted portion of the Type 1 font file after it has been decoded. |
| <i>len3</i> | Length in bytes of the portion of the Type 1 font file that contains the 512 zeros, plus the cleartomark operator, plus any following data. |
| <i>hasDW</i> | If <i>true</i> , the <i>dw</i> and <i>w</i> fields are used; if <i>false</i> , they are not used. |

dw

(Optional) Default width for glyphs in a CIDFont. See Section 7.7.9, "CIDFontType 0," in the [Portable Document Format Reference Manual](#) for more information.

w

A Cos array of a set of lists that define the widths for the glyphs in the CIDFont. Each list can specify individual widths for consecutive CIDs, or one width for a range of CIDs. See Section 7.7.11, "Character widths in CIDFonts," in the [Portable Document Format Reference Manual](#) for information on the format of this array.

hasDW2

If *true*, the *dw2* and *w2* fields are used; if *false*, they are not used.

dw2

(Optional; applies only to CIDFonts that are used for vertical writing) The default metrics for writing mode 1. This entry is an array of two numbers: the *y* component of the position vector and the *y* component of the displacement vector for writing mode 1. The *x* component of the position vector is always half the width of the character. The *x* component of the displacement vector is always 0. The default value is [880 -1000]. For information on writing mode 1, see Section 7.7.12, "Vertical Writing," in the [Portable Document Format Reference Manual](#).

w2

(Optional; applies only to CIDFonts that are used for vertical writing) A Cos array defining the metrics for vertical writing. Its format is similar to the format of the array in *w*. It defines the *x* and *y* components of the position vector, and the *y* component of the displacement vector. The *x* component of the displacement vector is always 0. See Section 7.7.11, "Character widths in CIDFonts," in the [Portable Document Format Reference Manual](#) for information on the format of this array.

toUnicodeLen

(Optional) Length of *toUnicodeStm*.

toUnicodeStm

(Optional) A stream containing a CMap that defines the mapping from character codes to Unicode values. This entry is recommended for fonts that do not use one of the predefined CMaps. If present, this allows strings in the encoding to convert to Unicode values for export to other applications or plug-ins. For more information, see Section 7.7.7, "Type 0 Fonts," in the [Portable Document Format Reference Manual](#).

cidToGidMapStm

A stream contain the mapping from CID to glyphindex ("GID"). The glyphindex for a particular CID value c is a 2-byte value stored in bytes $2*c$ and $2*c+1$; the first byte is the high-order byte.

panoseNo

A 12-byte string containing the Family Class ID, Family SubClass ID, and 10 bytes for the Panose classification number for the font. For additional details on the Panose number, see *Japanese TrueType Font Property Selection Guidelines* by the TrueType Conference Technical Committee.

fd

A Cos dictionary identifying a subset of characters in a CIDFont. The values are dictionaries with entries that override the values in the **FontDescriptor** dictionary for the subset of characters. See Section 7.10.3, "Font descriptors for Type 0 fonts," in the [Portable Document Format Reference Manual](#) for more information.

cidSetStm

A stream identifying which CIDs are present in the CIDFont file. It is required if the CIDFont file is embedded and only contains a subset of the glyphs in the character collection defined by the CIDSystemInfo. If this entry is missing, then it is assumed that the CIDFont file contains all the glyphs for the character collection. The stream's length should be rounded up to the nearest multiple of 8. The bits should be stored in bytes with the high-order bit first. Each bit corresponds to a CID. The first bit of the first byte corresponds to CID 0, the next bit corresponds to CID 1, and so on. If the subset contains a CID, the bit for that CID should be set. For compactness, the stream may use one of the compression filters to encode the data. For more information, see Section 7.10.3, "Font descriptors for Type 0 fonts," in the [Portable Document Format Reference Manual](#).

flags

One of the [PDEFontCreateFlags](#) describing how to embed and subset the font.

mmDesignVec

Pointer to multiple master font design vector.

PDEFontInfoRec

PDEFontInfoRecP

```
typedef struct _t_PDEFontInfo {  
    ASAtom name;  
    ASAtom type;  
    ASAtom charSet;  
    ASAtom encoding;  
} PDEFontInfoRec, *PDEFontInfoRecP;
```

Description [PDEFont](#) information.

Header File *PSFExpT.h*

Related Methods [PDSysFontGetInfo](#)

| | |
|-----------------|--|
| <i>name</i> | An <i>ASAtom</i> for font name, as in "Times-Roman." |
| <i>type</i> | An <i>ASAtom</i> for font type, "Type 1," "TrueType," and so on. |
| <i>charSet</i> | An <i>ASAtom</i> for "Roman" or <i>ASAtomNull</i> . If "Roman," the characters must be a subset of the Adobe Standard Roman Character Set. |
| <i>encoding</i> | An <i>ASAtom</i> for font encoding, as in WinAnsiEncoding . |

PDEGraphicState

PDEGraphicStateP

```
typedef struct _t_PDEGraphicState {
    ASUns32 wasSetFlags;
    PDEColorSpec fillColorSpec;
    PDEColorSpec strokeColorSpec;
    PDEDash dash;
    ASFixed lineWidth;
    ASFixed miterLimit;
    ASFixed flatness;
    ASInt32 lineCap;
    ASInt32 lineJoin;
    ASAtom renderIntent;
    PDEExtGState extGState;
} PDEGraphicState, *PDEGraphicStateP;
```

| | |
|------------------------|---|
| Description | Attributes of a PDEElement or a PDEText subelement. |
| Header File | <i>PEExpT.h</i> |
| Related Methods | PDEDefaultGState PDETextAdd |

wasSetFlags

[PDEGraphicStateWasSetFlags](#) indicating if an attribute has been set.

Note: Support for these flags is not complete. For compatibility, you should set them, but do not depend on reading their values back. The intended use is with XObject Forms to indicate whether the value is inherited or explicitly set.

fillColorSpec

Fill color specification.

strokeColorSpec

Stroke color specification.

dash

Dash specification.

lineWidth

Line width, corresponding to the **w** ([setlinewidth](#)) operator.

miterLimit

Miter limit, corresponding to the **M** (**setmiterlimit**) operator.

flatness

Line flatness, corresponding to the **i** (**setflat**) operator.

lineCap

Line cap style, corresponding to the **J** (**setlinecap**) operator.

lineJoin

Line join style, corresponding to the **j** (**setlinejoin**) operator.

renderIntent

A color rendering intent, corresponding to the **Intent** key in the image dictionary.

extGState

An extended graphics, corresponding to the **gs** operator.

PDEGraphicStateWasSetFlags

```
typedef enum {
    kPDEFillCSpaceWasSet = 0x0001,
    kPDEFillCValueWasSet = 0x0002,
    kPDEStrokeCSpaceWasSet = 0x0004,
    kPDEStrokeCValueWasSet = 0x0008,
    kPDEDashWasSet = 0x0010,
    kPDELineWidthWasSet = 0x0020,
    kPDEMiterLimitWasSet = 0x0040,
    kPDEFlatnessWasSet = 0x0080,
    kPDELineCapWasSet = 0x0100,
    kPDELineJoinWasSet = 0x0200,
    kPDERenderIntentWasSet = 0x0400,
    kPDEExtGStateWasSet = 0x0800
} PDEGraphicStateWasSetFlags;
```

| | |
|------------------------|--|
| Description | Structure describing the graphics state that was set. |
| Header File | <i>PEExpT.h</i> |
| Related Methods | PDEDefaultGState PDETextAdd |

| |
|---|
| <i>kPDEFillCSpaceWasSet</i> |
| A fill color space was set, corresponding to the cs (setcolorspace) operator. |
| <i>kPDEFillCValueWasSet</i> |
| A color fill value was set, corresponding to the sc (setcolor) operator. |
| <i>kPDEStrokeCSpaceWasSet</i> |
| A color space stroke value was set, corresponding to the CS (setcolorspace) operator. |
| <i>kPDEStrokeCValueWasSet</i> |
| A color stroke value was set, corresponding to the SC (setcolor) operator. |
| <i>kPDEDashWasSet</i> |
| A dash specification was set, corresponding to the d (setdash) operator. |
| <i>kPDELineWidthWasSet</i> |
| The line width was set, corresponding to the w (setlinewidth) operator. |

kPDEMiterLimitWasSet

The miter limit was set, corresponding to the **M** (**setmiterlimit**) operator.

kPDEFlatnessWasSet

Line flatness was set, corresponding to the **i** (**setflat**) operator.

kPDELineCapWasSet

Line cap style was set, corresponding to the **J** (**setlinecap**) operator.

kPDELineJoinWasSet

Line join style was set, corresponding to the **j** (**setlinejoin**) operator.

kPDERenderIntentWasSet

A color rendering intent was set, corresponding to the **Intent** key in the image dictionary.

kPDEExtGStateWasSet

An extended graphics state was set, corresponding to the **gs** operator.

PDEGrayCalFIt

```
typedef AGMGrayCalFIt PDEGrayCalFIt;
```

| | |
|--------------------------|--|
| Description | Structure describing a CalGray color space. Same as AGMGrayCalFIt . |
| Header File | <i>PEExpt.h</i> |
| Related Types | AGMGrayCalFIt PDEColorSpaceStruct |
| Related Callbacks | None |
| Related Methods | PDEColorSpaceCreate |

PDEICCBasedColorData

```
typedef struct _t_PDEICCBasedColorData {  
    ASSize_t size;  
    ASStm iccstream;  
    ASUns32 nComps;  
    PDEColorSpace altCs;  
} PDEICCBasedColorData;
```

Description Structure describing an **ICCBased** color space.

Header File *PEExpt.h*

Related Types [PDEColorSpaceStruct](#)

Related Callbacks None

Related Methods [PDEColorSpaceCreate](#)

| | |
|------------------|--|
| <i>size</i> | Size of the data structure. Must be set to <i>sizeof(PDEICCBasedColorData)</i> . |
| <i>iccstream</i> | Stream containing ICC Profile. |
| <i>nComps</i> | Number of color components (1, 3, or 4). |
| <i>altCs</i> | (<i>Optional</i>) Alternate ColorSpace. |

PDEImageAttrFlags

```
typedef enum {
    kPDEImageExternal = 0x0001,
    kPDEImageIsMask = 0x0002,
    kPDEImageInterpolate = 0x0004,
    kPDEImageHaveDecode = 0x0008,
    kPDEImageIsIndexed = 0x0010
    /* Added in Acrobat 4.0 */
    kPDEImageMaskedByPosition,
    kPDEImageMaskedByColor
} PDEImageAttrFlags;
```

Description Flags for [PDEImageAttrs](#). See Section 7.12.1 in the [Portable Document Format Reference Manual](#) for more information on image attributes.

Header File *PEExpT.h*

Related Methods [PDEImageCreate](#)
[PDEImageGetAttrs](#)
[PDEImageGetData](#)

| | |
|----------------------------------|--|
| <i>kPDEImageExternal</i> | Indicates image is an <i>XObject</i> . |
| <i>kPDEImageIsMask</i> | Indicates image is an imagemask. |
| <i>kPDEImageInterpolate</i> | Indicates interpolate is <i>true</i> . |
| <i>kPDEImageHaveDecode</i> | Indicates there is a decode array. |
| <i>kPDEImageIsIndexed</i> | Indicates image uses an indexed color space. |
| <i>kPDEImageMaskedByPosition</i> | Indicates image has a Mask key containing an ImageMask stream. |
| <i>kPDEImageMaskedByColor</i> | Indicates image has a Mask key containing an array of color values. |

PDEImageAttrs

PDEImageAttrsP

```
typedef struct _t_PDEImageAttrs {
    ASUns32 flags;
    ASInt32 width;
    ASInt32 height;
    ASInt32 bitsPerComponent;
    ASFixed decode[8];
    /* New in Acrobat 4.0 */
    ASAtom intent;
} PDEImageAttrs, *PDEImageAttrsP;
```

| | |
|------------------------|---|
| Description | Attributes of a PDEImage . |
| Header File | <i>PEExpT.h</i> |
| Related Methods | PDEImageCreate PDEImageGetAttrs PDEImageGetData |

| | |
|-------------------------|---|
| <i>flags</i> | PDEImageAttrFlags indicating image attributes. |
| <i>width</i> | Width of the image, corresponding to the Width key in the image dictionary. |
| <i>height</i> | Height of the image, corresponding to the Height key in the image dictionary. |
| <i>bitsPerComponent</i> | Number of bits used to represent each color component in the image, corresponding to the BitsPerComponent key in the image dictionary. |
| <i>decode</i> | An array of numbers specifying the mapping from sample values in the image to values appropriate for the current color space. These values correspond to the Decode key in the image dictionary. |

intent

Color rendering intent, corresponding to the **Intent** key in the image dictionary.

PDEImageDataFlags

```
typedef enum {  
    kPDEImageEncodedData = 0x0001  
} PDEImageDataFlags;
```

Description Flags for [PDEImageGetData](#), [PDEImageGetDataStm](#), [PDEImageSetData](#), and [PDEImageSetDataStm](#).

Header File *PEExpT.h*

Related Methods [PDEImageGetData](#)
[PDEImageGetDataStm](#)
[PDEImageSetData](#)
[PDEImageSetDataStm](#)

kPDEImageEncodedData

Indicates filter is active; data is encoded.

PDEIndexedColorData

```
typedef struct _t_PDEIndexedColorData {  
    ASSize_t size;  
    PDEColorSpace baseCs;  
    ASUns16 hival;  
    char* lookup;  
    ASUns32 lookupLen;  
} PDEIndexedColorData;
```

Description Structure describing an **Indexed** color space.

Header File *PEExpt.h*

Related Types [PDEColorSpaceStruct](#)

Related Callbacks None

Related Methods [PDEColorSpaceCreate](#)

| | |
|------------------|---|
| <i>size</i> | Size of the data structure. Must be set to <i>sizeof(PDEIndexedColorData)</i> . |
| <i>baseCs</i> | Base colorspace. |
| <i>hival</i> | Highest color value. |
| <i>lookup</i> | Indexed color lookup data. |
| <i>lookupLen</i> | Number of bytes in lookup data. |

PDELabCalFlt

```
typedef AGMLabCalFlt PDELabCalFlt;
```

| | |
|--------------------------|--|
| Description | Structure describing a L*a*b* color space. Same as AGMLabCalFlt . |
| Header File | <i>PEExpt.h</i> |
| Related Types | AGMLabCalFlt PDEColorSpaceStruct |
| Related Callbacks | None |
| Related Methods | PDEColorSpaceCreate |

PDEPathElementType

```
typedef enum {
    kPDEMoveTo,
    kPDELineTo,
    kPDECurveTo,
    kPDECurveToV,
    kPDECurveToY,
    kPDERect,
    kPDEClosePath,
    kPDEPathLastType
} PDEPathElementType;
```

| | |
|------------------------|--|
| Description | Enumerated data type for path segment operators in PDEPath elements. |
| Header File | <i>PEExpT.h</i> |
| Related Methods | PDEPathAddSegment PDEPathCreate PDEPathSetData |

| |
|--|
| <i>kPDEMoveTo</i> |
| Designates m (moveto) operator, which moves the current point. |
| <i>kPDELineTo</i> |
| Designates l (lineto) operator, which appends a straight line segment from the current point. |
| <i>kPDECurveTo</i> |
| Designates c (curveto) operator, which appends a Bézier curve to the path. |
| <i>kPDECurveToV</i> |
| Designates v (curveto) operator, which appends a Bézier curve to the current path when the first control point coincides with initial point on the curve. |
| <i>kPDECurveToY</i> |
| Designates y (curveto) operator, which appends a Bézier curve to the current path when the second control point coincides with final point on the curve. |
| <i>kPDERect</i> |
| Designates re operator, which adds a rectangle to the current path. |

kPDEClosePath

Designates **h (closepath)** operator, which closes the current subpath.

PDEPathOpFlags

```
typedef enum {  
    kPDEInvisible = 0x00,  
    kPDEStroke = 0x01,  
    kPDEFill = 0x02,  
    kPDEEoFill = 0x04  
} PDEPathOpFlags;
```

Description Flags for paint operators in a [PDEPath](#).

Header File *PEExpT.h*

Related Methods [PDEPathGetPaintOp](#)
[PDEPathSetPaintOp](#)

| |
|--|
| <i>kPDEInvisible</i> |
| Path is neither stroked nor filled, so it is invisible. |
| <i>kPDEStroke</i> |
| Stroke the path, as with the S (stroke) operator. |
| <i>kPDEFill</i> |
| Fills the path, using the nonzero winding number rule to determine the region to fill, as with the f (fill) operator. |
| <i>kPDEEoFill</i> |
| Fills the path, using the even-odd rule to determine the region to fill, as with the f* (eofill) operator. |

PDEPatternColorSpace

```
typedef PDEColorSpace PDEPatternColorSpace;
```

| | |
|--------------------------|---|
| Description | <i>PDEColorSpace</i> describing a Pattern color space. |
| Header File | <i>PEExpt.h</i> |
| Related Types | PDEColorSpaceStruct |
| Related Callbacks | None |
| Related Methods | PDEColorSpaceCreate |

PDERGBCalFlt

```
typedef AGMRGBCalFlt PDERGBCalFlt;
```

| | |
|--------------------------|--|
| Description | Structure describing a CalRGB color space. Same as AGMRGBCalFlt . |
| Header File | <i>PEExpt.h</i> |
| Related Types | AGMRGBCalFlt PDEColorSpaceStruct |
| Related Callbacks | None |
| Related Methods | PDEColorSpaceCreate |

PDESeparationColorData

```
typedef struct _t_PDESeparationColorData {  
    ASSize_t size;  
    ASAtom name;  
    PDEColorSpace alt;  
    CosObj tintTransform;  
} PDESeparationColorData;
```

Description Structure describing a **Separation** color space.

Header File *PEExpt.h*

Related Types [PDEColorSpaceStruct](#)

Related Callbacks None

Related Methods [PDEColorSpaceCreate](#)

size

Size of the data structure. Must be set to *sizeof(PDESeparationColorData)*.

name

Name of separation or colorant.

alt

Alternative colorspace.

tintTransform

The tintTransform dictionary or function. See Section 7.11 in the [Portable Document Format Reference Manual](#).

PDSysFontPlatData

PDSysFontPlatDataP

```
/* In Windows */
typedef struct _t_PDSysFontPlatData {
    ASSize_t size;
    LOGFONT* lf;
    ASPathName fontPath;
    ASPathName afmPath;
} PDSysFontPlatData, *PDSysFontPlatDataP;

/* In Mac OS */
typedef struct _t_PDSysFontPlatData {
    ASSize_t size;
    ASInt16 fontID;
    ASInt16 fontStyle;
} PDSysFontPlatData, *PDSysFontPlatDataP;

/* In UNIX */
typedef struct _t_PDSysFontPlatData {
    ASSize_t size;
    ASPathName fontPath;
    ASPathName afmPath;
} PDSysFontPlatData, *PDSysFontPlatDataP;
```

| | |
|------------------------|--|
| Description | System font platform-dependent data. |
| Header File | <i>PDSysFontExpT.h</i> |
| Related Methods | PDSysFontAcquirePlatformData PDSysFontReleasePlatformData |

| | |
|-----------------|--|
| <i>size</i> | Size of the data structure. Must be set to <i>sizeof(PDSysFontPlatData)</i> . |
| <i>If</i> | (<i>Windows only</i>) Windows <i>LOGFONT</i> structure defining font attributes. |
| <i>fontPath</i> | (<i>Optional for Windows</i>) A path to the font file. Set only if <i>If</i> is not present. |

afmPath

(Optional for Windows) A path to the font AFM file. Set only if *lf* is not present.

fontID

Macintosh FOND id.

fontStyle

Macintosh style value within that Fond. Default is 0.

PDETextFlags

```
typedef enum {  
    kPDETextRun = 0x0001,  
    kPDETextChar = 0x0002,  
    kPDETextPageSpace = 0x0004,  
    kPDETextGetBounds = 0x0008  
} PDETextFlags;
```

Description Bit field used in [PDEText](#) methods.

Header File *PEExpT.h*

Related Methods [PDETextAdd](#)
[PDETextGetFont](#)
[PDETextGetText](#)

| |
|---|
| <i>kPDETextRun</i> |
| Text run. |
| <i>kPDETextChar</i> |
| Character (text run with only one character). |
| <i>kPDETextPageSpace</i> |
| Obtain the advance width in page space. |
| <i>kPDETextGetBounds</i> |
| Fill in the left and right bounds of the text run's bounding box. |

PDETextRenderMode

```
typedef enum {
    kPDETextFill,
    kPDETextStroke,
    kPDETextFillAndStroke,
    kPDETextInvisible
} PDETextRenderMode;
```

| | |
|------------------------|---|
| Description | Flags indicating text rendering mode set by the Tr operator. |
| Header File | <i>PEExpT.h</i> |
| Related Methods | PDETextCreate |

| |
|--|
| <i>kPDETextFill</i> |
| Fill text. |
| <i>kPDETextStroke</i> |
| Stroke text. |
| <i>kPDETextFillAndStroke</i> |
| Fill and stroke text. |
| <i>kPDETextInvisible</i> |
| Text with no fill and no stroke (invisible). |

PDETextState

PDETextStateP

```
typedef struct _t_PDETextState {
    ASUns32 wasSetFlags;
    ASFixed charSpacing;
    ASFixed wordSpacing;
    ASInt32 renderMode;
} PDETextState, *PDETextStateP;
```

| | |
|------------------------|---|
| Description | Attributes of a PDEText element. |
| Header File | <i>PEExpT.h</i> |
| Related Methods | PDETextAdd PDETextGetTextState PDETextRunSetTextState |

wasSetFlags

[PDETextStateWasSetFlags](#) indicating if an attribute has been set.

charSpacing

Character spacing was set, corresponding to the **Tc** operator.

wordSpacing

Word spacing, corresponding to the **Tw** operator.

renderMode

Text rendering mode, corresponding to the **Tr** operator.

PDETextStateWasSetFlags

```
typedef enum {  
    kPDECharSpacingWasSet = 0x0001,  
    kPDEWordSpacingWasSet = 0x0002,  
    kPDERenderModeWasSet = 0x0004  
} PDETextStateWasSetFlags;
```

Description Structure describing the text state that was set.

Header File *PEExpT.h*

Related Methods [PDETextAdd](#)
 [PDETextGetTextState](#)
 [PDETextRunSetTextState](#)

| |
|---|
| <i>kPDECharSpacingWasSet</i> |
| Character spacing was set, corresponding to the Tc operator. |
| <i>kPDEWordSpacingWasSet</i> |
| Word spacing was set, corresponding to the Tw operator. |
| <i>kPDERenderModeWasSet</i> |
| Text rendering mode was set, corresponding to the Tr operator. |

PDEType

```
typedef enum {
    kPDEContent,
    kPDEText,
    kPDEPath,
    kPDEImage,
    kPDEForm,
    kPDEPS,
    kPDEXObject,
    kPDEClip,
    kPDEFont,
    kPDEColorSpace,
    kPDEExtGState,
    kPDEPlace,
    kPDEContainer,
    kPDSysFont,
    kPDEPattern,
    /* Added in Acrobat 4.0 */
    kPDEDeviceNCColors,
    kPDEShading,
    kPDEGroup,
    kPDEUnknown,
    kPDELastType
} PDEType;
```

Description Types of [PDEObject](#), which is the superclass for [PDEContent](#), [PDEElement](#), [PDEClip](#), and so on.

Header File *PEExpT.h*

Related Methods [PDEObjectGetType](#)

kPDEContent

[PDEContent](#) object.

kPDEText

[PDEText](#) object.

kPDEPath

[PDEPath](#) object.

kPDEImage

[PDEImage](#) object.

| |
|--|
| <i>kPDEForm</i> PDEForm object. |
| <i>kPDEXObject</i> PDEXObject object. |
| <i>kPDEClip</i> PDEClip object. |
| <i>kPDEFont</i> PDEFont object. |
| <i>kPDEColorSpace</i> PDEColorSpace object. |
| <i>kPDEExtGState</i> PDEExtGState object. |
| <i>kPDEPlace</i> PDEPlace object. |
| <i>kPDEContainer</i> PDEContainer object. |
| <i>kPDSSysFont</i> PDSysFont object. |
| <i>kPDEPattern</i> PDEPattern object. |
| <i>kPDEDeviceNColors</i> PDEDeviceNColors object. |
| <i>kPDEShading</i> PDEShading object. |
| <i>kPDEGroup</i> PDEGroup object. |
| <i>kPDEUnknown</i> PDEUnknown object. |

PDFFileSpecHandler

```
typedef struct _t_PDFFileSpecHandler {  
    ASSize_t size;  
    PDFFileSpecNewFromASPathProc NewFromASPath;  
    PDFFileSpecAcquireASPathProc AcquireASPath;  
    /* New for Acrobat 3.0 */  
    PDLaunchActionProc LaunchAction;  
    /* Added in Acrobat 4.0 */  
    ASFileSys fileSys;  
} PDFFileSpecHandlerRec, *PDFFileSpecHandler;
```

Description Data structure that implements a file specification handler. It contains callbacks that implement the filespec handler's functions (converting from a file specification to an [ASPathName](#), creating a new file specification from an [ASPathName](#), and launching the specified file).

Header File *PDExpT.h*

Related Methods [PDRegisterFileSpecHandler](#)

size

Size of the data structure. Must be set to *sizeof(PDFFileSpecHandlerRec)*.

fileSys

The file system that is used to read data for this file specification handler.

PDFLData

```
/* In Windows */

typedef struct _t_PDFLData {
    ASSize_t size;
    ASUns32 flags;
    HINSTANCE inst;
    char** dirList;
    ASInt32 listLen;
    TKAllocatorProcs allocator;
    TKResourceProcs resProcs;
} PDFLDataRec, *PDFLData;

/* In Mac OS */

typedef struct _t_PDFLData {
    ASSize_t size;
    ASUns32 flags;
    FSSpec** dirList;
    ASInt32 listLen;
    TKAllocatorProcs allocator;
    short resFile;
    TKResourceProcs resProcs;
} PDFLDataRec, *PDFLData;

/* In UNIX */

typedef struct _t_PDFLData {
    ASSize_t size;
    ASUns32 flags;
    char** dirList;
    ASInt32 listLen;
    TKAllocatorProcs allocator;
    TKResourceProcs resProcs;
} PDFLDataRec, *PDFLData;
```

| | |
|------------------------|--|
| Description | (Present only in Adobe PDF Library 1.0 or later) Structure for PDFLInit . |
| Header File | <i>PDFInit.h</i> |
| Related Methods | PDFLInit |

size

Size of the data structure. Must be set to *sizeof(PDFLDataRec)*.

flags

Currently unused. Should be set to 0.

inst (Windows only)

Location of resources for the Library. For a static library, you link them into your application and use *NULL* for *inst*. For a DLL version of the library, call *LoadLibrary* on the DLL and set *inst* equal to the *HINSTANCE* returned:

```
libData.inst = LoadLibrary("PlacedPDF.DLL");
```

If you are linking the static library into a DLL, pass the *HINSTANCE* passed into *DllMain* or *WinMain* for *inst* and link the resources into the DLL.

Free the *HINSTANCE* with *FreeLibrary* after you call [PDFLTerm](#).

dirList

List of directories with fonts. Here is an example for Windows:

```
libData.listLen = 2L;
libData.dirList = (ASPathName *)malloc(sizeof
    (ASPathName)*2);
libData.dirList[0]= (ASPathName)"c:\\PSFonts";
libData.dirList[1]=(ASPathName)"c:\\Fonts";
```

resFile (Macintosh only)

Resources location, if they are not in the application itself. Set them using *OpenResFile*:

```
libData.resFile = OpenResFile("\\pMyResFileGoesHere").
```

listLen

Number of directories listed in *dirList*.

allocator

The [TKAllocatorProcs](#) structure containing function pointers for memory allocation callbacks. The library manages its own memory. Most applications linking with the library should pass *NULL* for this member. Applications wishing more control over memory allocation can fill out this struct and will be responsible for providing and freeing memory as well as reporting available memory as requested by the Library.

resProcs

The [TKResourceProcs](#) structure containing function pointers for resource management callbacks. The library manages its own resources. Most applications linking with the library should pass *NULL* for this member. Applications wishing more control over resource allocation can fill out this struct and will be responsible for providing and freeing resources requested by the Library.

PDFLPrintUserParamsRec

```
/* In UNIX */
typedef struct {
    ASInt32 size;
    PDPrintParams printParams;
    ASBool emitToFile;
    ASStm printStm;
    ASUnsl6 paperWidth;
    ASUnsl6 paperHeight;
    ASInt32 dontEmitListLen;
    char** dontEmitList;
    ASBool emitToPrinter;
    char* command;
    PDFLPrintCancelProc cancelProc;
    void* clientData;
} PDFLPrintUserParamsRec, *PDFLPrintUserParams;

/* In Windows */
typedef struct {
    ASInt32 size;
    ASBool emitToFile;
    PDPrintParams printParams;
    ASStm printStm;
    ASUnsl6 paperWidth;
    ASUnsl6 paperHeight;
    ASInt32 dontEmitListLen;
    char** dontEmitList;
    ASBool emitToPrinter;
    char* inFileName;
    char* outFileName;
    char* deviceName;
    char* driverName;
    char* portName;
    DEVMODE* pDevMode;
    int    startPage;
    int    endPage;
    int    shrinkToFit;
    int    printAnnots;
    int    psLevel;
    int    nCopies;
    int    binaryOK;
    int    emitHalftones;
    int    reverse;
    int    farEastFontOpt;
    PDFLPrintCancelProc cancelProc;
```

```
void* clientData;
} PDFLPrintUserParamsRec, *PDFLPrintUserParams;

/* In Mac OS */

typedef struct {
    ASInt32 size;
    ASBool emitToFile;
    PDPrintParams printParams;
    ASStm printStm;
    ASUns16 paperWidth;
    ASUns16 paperHeight;
    ASInt32 dontEmitListLen;
    char** dontEmitList;
    PDFLPrintCancelProc cancelProc;
    void* clientData;
} PDFLPrintUserParamsRec, *PDFLPrintUserParams;
```

| | |
|--------------------------|---|
| Description | Used To control printing with PDFLPrintDoc . |
| Header File | <i>PDFLPrint.h</i> |
| Related Callbacks | PDFLPrintCancelProc |
| Related Methods | PDDocPrintPages PDFLPrintDoc PDFLPrintPDF |

| For all platforms | |
|--------------------|--|
| <i>size</i> | Size of the data structure. Must be set to <code>sizeof(PDFLPrintUserParamsRec)</code> . |
| <i>printParams</i> | PDPrintParams structure. |
| <i>emitToFile</i> | Create a PostScript file? |
| <i>printStm</i> | Writable <i>ASStm</i> that points to file stm or proc stm. |
| <i>paperWidth</i> | Width of paper in points—default is 612. |
| <i>paperHeight</i> | Height of paper in points—default is 792. |

| For all platforms |
|---|
| <p><i>dontEmitListLen</i> Number of fonts that should not be downloaded.</p> |
| <p><i>dontEmitList</i> List of fonts (T1, TT, CID) that should not be downloaded.</p> |
| <p><i>emitToPrinter</i> Output PDF file to a printer .</p> |
| <p><i>cancelProc</i> Optional PDFLPrintCancelProc.</p> |
| <p><i>clientData</i> Optional data passed to <i>cancelProc</i>.</p> |
| <p><i>binaryOK</i> PostScript printing only — 1 turns on binary data, 0 turns it off.</p> |
| <p><i>emitHalftones</i> PostScript printing only — 1 turns on emitHalftones, 0 turns it off.</p> |
| <p><i>reverse</i> PostScript printing only — 1 reverses print order, 0 does regular order.</p> |
| <p><i>farEastFontOpt</i> PostScript printing only — See PDFFarEastFont enum in <i>PDPrint.h</i>. Default is PDPDFarEastFont_Download_All.</p> |
| In UNIX |
| <p><i>command (Unix only)</i> Optional command line arguments, used only if <i>emitToPrinter</i> is true. For example, "lp" or "lpr."</p> |
| In Windows |
| <p><i>inFileName</i> Used for the Windows DOCINFO structure — <i>lpzDocName</i> that points to a <i>NULL</i>-terminated string that specifies the name of the document.</p> |

| In Windows | |
|--------------------|--|
| <i>outFileName</i> | Used for the Windows DOCINFO structure — <i>lpOutput</i> . It points to a <i>NULL</i> -terminated string that specifies the name of an output file. If this pointer is <i>NULL</i> , the output will be sent to the device identified by <i>deviceName</i> , <i>driverName</i> , and <i>portName</i> . |
| <i>deviceName</i> | Name of device to print to. For example, "Distiller Assistant v3.01." Available devices can be found in the Windows Registry at HKEY_CURRENT_USER\Software\Microsoft\WindowsNT\CurrentVersion\Devices |
| <i>driverName</i> | Name of driver. For example, "winspool." See registry key for <i>deviceName</i> . |
| <i>portName</i> | Name of Port. For example, "LPT1." See registry key for <i>deviceName</i> . |
| <i>pDevMode</i> | (Optional) Allows you to pass a DEVMODE structure in if you have one. |
| <i>startPage</i> | Page to start printing with, 0-based. |
| <i>endPage</i> | Page to finish printing on. |
| <i>shrinkToFit</i> | 1 turns on shrinkToFit, 0 turns it off. |
| <i>printAnnots</i> | 1 turns on printing annots, 0 turns it off. |
| <i>psLevel</i> | PostScript printing only — PostScript level. |
| <i>nCopies</i> | PostScript printing only — number of copies to print. |

PDFontEncoding

| | |
|------------------------|--|
| Description | Enumerated data type. Specifies a font's encoding. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDFontGetEncodingIndex |

PDFBuiltInEncoding

The encoding specified internally in the font. In the case of a Type 1 or MMType 1 font, this is specified by the **Encoding** value in the font's fontdict. In the case of TrueType fonts, this is the encoding specified by the default one-byte cmap for the platform.

PDMacRomanEncoding

MacRomanEncoding, as defined in Appendix C in the [Portable Document Format Reference Manual](#).

PDMacExpertEncoding

MacExpertEncoding, as defined in Appendix C in the [Portable Document Format Reference Manual](#).

PDWinAnsiEncoding

WinAnsiEncoding, as defined in Appendix C in the [Portable Document Format Reference Manual](#).

PDStdEncoding

StandardEncoding, as defined in Appendix C in the [Portable Document Format Reference Manual](#).

PDFDocEncoding

PDFDocEncoding, as defined in Appendix C in the [Portable Document Format Reference Manual](#). This will never be returned for a font; it is used internally.

PDFontMetrics

PDFontMetricsP

```
typedef struct _t_PDFontMetrics {
    ASUns32 flags;
    ASFixedRect fontBBox;
    ASInt16 missingWidth;
    ASInt16 stemV;
    ASInt16 stemH;
    ASInt16 capHeight;
    ASInt16 xHeight;
    ASInt16 ascent;
    ASInt16 descent;
    ASInt16 leading;
    ASInt16 maxWidth;
    ASInt16 avgWidth;
    ASInt16 italicAngle;
    /* Added in Acrobat 4.0 */
    PDFontStyles style;
    ASInt16 baseLineAdj;
} PDFontMetrics, *PDFontMetricsP;
```

| | |
|------------------------|---|
| Description | Data structure containing information about a font's metrics. See Section 2.3.7 in the Portable Document Format Reference Manual for more information about font metrics. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDFontGetMetrics PDFontSetMetrics |

flags

Must be an OR of the [Font Flags](#) values. All unused flags must be off.

fontBBox

Font bounding box in 1000 EM units. (An EM is a typographic unit of measurement equal to the size of a font. In a 12-point font, an EM is 12 points.)

missingWidth

The width of missing char (*.notdef*).

| | |
|--------------------|--|
| <i>stemV</i> | The vertical stem width. |
| <i>stemH</i> | The horizontal stem width. |
| <i>capHeight</i> | The capital height. |
| <i>xHeight</i> | The X height. |
| <i>ascent</i> | The maximum ascender height. |
| <i>descent</i> | The maximum descender depth. |
| <i>leading</i> | The additional leading between lines. |
| <i>maxWidth</i> | The maximum character width. |
| <i>avgWidth</i> | The average character width. |
| <i>italicAngle</i> | The italic angle in degrees, if any. |
| <i>style</i> | Panose and sFamily class values. |
| <i>baseLineAdj</i> | Baseline adjustment, which is a vertical adjustment for font baseline difference and writing mode 1 (vertical). This should only be used for CIDFontType 2 fonts with font substitution. See Section 7.7.10, "CIDFontType 2," in the Portable Document Format Reference Manual for more information. |

PDFontStyles

```
typedef struct _t_PDFontStyles {  
    ASUns8 sFamilyClassID;  
    ASUns8 sFamilySubclassID;  
    ASUns8 bFamilyType;  
    ASUns8 bSerifStyle;  
    ASUns8 bWeight;  
    ASUns8 bProportion;  
} PDFontStyles;
```

Description Data structure containing Panose and sFamily class values for a font. See Section 7.10.3, "Font descriptors for CID fonts," in the [Portable Document Format Reference Manual](#) for more information. For additional details on the Panose number, see *Japanese TrueType Font Property Selection Guidelines* by the TrueType Conference Technical Committee.

Header File *PDExpT.h*

Related Methods [PDFontGetMetrics](#)
[PDFontSetMetrics](#)

| |
|---|
| <i>sFamilyClassID</i> |
| Number that identifies the font family and determines the meaning of the remaining Panose digits. Possible families are Latin, Kanji, Hebrew, and so forth. |
| <i>sFamilySubclassID</i> |
| Number to identify the kind of family: text, decorative, handwritten, symbols, and so on. |
| <i>bFamilyType</i> |
| Number to identify the family type: text, decorative, handwritten, symbols, and so on. |
| <i>bSerifStyle</i> |
| Number that specifies the font's serif style, such as cove, obtuse cove, square, bone, and so forth. |
| <i>bWeight</i> |
| Number that specifies the font's weight, such as very light, heavy, black, and so on. |

bProportion

Number that specifies the font's proportions, such as modern, expanded, condensed, monospaced, and so on.

PDGraphicEnumMonitor

```
typedef struct _t_PDGraphicEnumMonitor {
    ASSize_t size;
    PDGraphicEnumTextProc EnumText;
    PDGraphicEnumPathProc EnumPath;
    PDGraphicEnumImageProc EnumImage;
    PDGraphicEnumXObjectRefProc EnumXObjectRef;
    PDGraphicEnumSaveProc EnumSave;
    PDGraphicEnumRestoreProc EnumRestore;
    PDGraphicEnumCharWidthProc EnumCharWidth;
    PDGraphicEnumCacheDeviceProc EnumCacheDevice;
    /* New for Acrobat 3.0 */
    PDGraphicEnumXObjectRefMatrixProc EnumXObjectRefMatrix;
} PDGraphicEnumMonitorRec, *PDGraphicEnumMonitor;
```

Description

An array of callbacks to pass to [PDPageEnumContents](#). One of the callbacks is called for every renderable object in the page contents. Pass *NULL* for a callback to *not* enumerate that type of object. Each array element must be either *NULL* or a valid callback. Enumeration of the page contents halts if the callback returns *false*.

Note: [PDPageEnumContents](#) is provided only for backwards compatibility. It has not been updated beyond PDF Version 1.1 and may not work correctly for newly created PDF 1.2 or later files. You should use the PDFEdit API to enumerate page contents.

Note: In versions at least through Acrobat 2.1, enumeration does not stop even if a method returns false.

Header File

PDExpT.h

Related Methods

[PDCharProcEnum](#)
[PDFormEnumPaintProc](#)
[PDPageEnumContents](#)

size

Size of the data structure. Must be set to `sizeof(PDGraphicEnumMonitorRec)`.

PDGraphicState

PDGraphicStateP

```
typedef struct _t_PDGraphicState {  
    ASFixedMatrix ctm;  
    ASFixed fillColor[4];  
    ASFixed strokeColor[4];  
    ASAtom fillCSpace;  
    ASAtom strokeCSpace;  
    ASFixed flatness;  
    ASInt32 lineCap;  
    ASFixed dashPhase;  
    ASInt32 dashLen;  
    ASFixed dashes[10];  
    ASInt32 lineJoin;  
    ASFixed lineWidth;  
    ASFixed miterLimit;  
} PDGraphicState, *PDGraphicStateP;
```

| | |
|------------------------|---|
| Description | Data structure containing information about the current graphics state. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDGraphicGetState |

PDImageAttrs

PDImageAttrsP

```
typedef struct _t_PDImageAttrs {
    ASInt32 width;
    ASInt32 height;
    ASInt32 bitsPerComponent;
    ASBool imageMask;
    ASBool interpolate;
    ASBool haveDecode;
    ASFixed decode[8];
    ASAtom colorSpaceName;
    ASBool isIndexed;
    ASInt32 hiVal;
    CosObj colorSpace;
    ASInt32 dataLen;
    /* Added in Acrobat 4.0 */
    ASInt32 comps;
} PDImageAttrs, *PDImageAttrsP;
```

Description Data structure containing information about the attributes of an image. See Section 7.12.1, "Images," in the [Portable Document Format Reference Manual](#) for more information.

Header File *PDExpT.h*

Related Methods [PDImageGetAttrs](#)
[PDInlineImageGetAttrs](#)

| |
|--|
| <i>width</i> (Required) Width of the source image in samples. |
| <i>height</i> (Required) Height of the source image in samples. |
| <i>bitsPerComponent</i> (Required) The number of bits used to represent each color component. |
| <i>imageMask</i> (Optional) <i>true</i> if the image should be treated as a mask; <i>false</i> otherwise. |

| |
|--|
| <i>interpolate</i> (Optional) <i>true</i> if interpolation is performed; <i>false</i> otherwise. Interpolation attempts to smooth transitions between sample values. |
| <i>haveDecode</i> <i>true</i> if <i>decode</i> is used; <i>false</i> otherwise. |
| <i>decode</i> (Optional) An array of numbers specifying the mapping from sample values in the image to values appropriate for the current color space. |
| <i>colorSpaceName</i> ASAtom representing the color space name. |
| <i>isIndexed</i> <i>true</i> if the color space is indexed; <i>false</i> otherwise. |
| <i>hiVal</i> (Optional) Used if <i>isIndexed</i> is <i>true</i> . Colors are specified by integers in the range 0 to <i>hiVal</i> . |
| <i>colorSpace</i> (Required for images, not allowed for image masks) Cos object of the color space used for the image samples. |
| <i>dataLen</i> Length of sample data, in bytes. |
| <i>comps</i> Number of components in <i>colorSpace</i> . For instance, <i>comps</i> is 3 for an RGB color space. |

PDInclusion

```
typedef AEnum8 PDInclusion;  
enum {  
    kIncludeOncePerDoc,  
    kIncludeOnEveryPage,  
    kIncludeNever  
};
```

Description Specifies how to include a resource in a file.

Header File *PDPrint.h*

Related Types [PDPrintParams](#)

Related Methods None

| |
|---|
| <i>kIncludeOncePerDoc</i> |
| Include the resource only once per file. |
| <i>kIncludeOnEveryPage</i> |
| Include the resource on every page in the file. |
| <i>kIncludeNever</i> |
| Never include the resource. |

PDLayoutMode

```
typedef AEnum8 PDLayoutMode;

enum {
    PDLayoutDontCare,
    PDLayoutSinglePage,
    PDLayoutOneColumn,
    PDLayoutTwoColumnLeft,
    PDLayoutTwoColumnRight
};
```

Description Structure that defines the layout of a document. The layout can be set as the viewer's *avpPageViewLayoutMode* preference (set by [AVAppSetPreference](#)) or in a view of a document by the *pageViewLayoutMode* field in [AVDocViewDef](#) (set by [AVDocGetViewDef](#)).

Header File *PDExpT.h*

Related Methods [AVDocGetViewDef](#)
[AVPageViewGetLayoutMode](#)
[AVPageViewSetLayoutMode](#)

PDLayoutDontCare

(Default) Use the user preference when opening the file, as specified in the *avpPageViewLayoutMode* preference, set by [AVAppSetPreference](#).

PDLayoutSinglePage

Use single page mode, as in pre-Acrobat 3.0 viewers.

PDLayoutOneColumn

Use one column continuous mode.

PDLayoutTwoColumnLeft

Use two column continuous mode with first page on left.

PDLayoutTwoColumnRight

Use two column continuous mode with first page on right.

PDLinkAnnotBorder

```
typedef struct _t_PDLinkAnnotBorder {  
    ASInt32 hCornerRadius;  
    ASInt32 vCornerRadius;  
    ASInt32 width;  
    ASInt32 dashArrayLen;  
    ASFixed dashArray[PDAnnotMaxDashes];  
} PDLinkAnnotBorder;
```

| | |
|------------------------|--|
| Description | Data structure specifying the attributes of a link annotation's border. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDLinkAnnotGetBorder PDLinkAnnotSetBorder |

PDOperation

| | |
|--------------------|--|
| Description | Enumerated data type. Specifies the type of changes that occurred for the PDDocWillChangePages and PDDocDidChangePages notifications. Not all <i>Did</i> notifications have corresponding <i>Will</i> notifications. The exceptions are listed in the table. |
| Header File | <i>PDExpT.h</i> |
| Methods | PDDocDeletePages PDPageSetRotate PDPageSetMediaBox PDPageSetCropBox |

| |
|--|
| <i>pdOpInsertPages</i> Page insertion. |
| <i>pdOpDeletePages</i> Page deletion. |
| <i>pdOpMovePages</i> Page rearrangement. |
| <i>pdOpRotatePages</i> Page rotation. |
| <i>pdOpCropPages</i> Page cropping. |
| <i>pdOpAddResource</i> Only PDDocDidChangePages exists, not PDDocWillChangePages . |
| <i>pdOpRemoveResource</i> Only PDDocDidChangePages exists, not PDDocWillChangePages . |
| <i>pdOpAddContents</i> Only PDDocDidChangePages exists, not PDDocWillChangePages . |
| <i>pdOpRemoveContents</i> Only PDDocDidChangePages exists, not PDDocWillChangePages . |
| <i>pdOpSetMediaBox</i> Page media box modification. |

PDOutputType

```
typedef AEnum8 PDOutputType;  
enum {  
    PDOutput_PS,  
    PDOutput_EPSNoPrev,  
    PDOutput_EPSMacStdPrev,  
    PDOutput_EPSMacExtPrev  
};
```

| | |
|------------------------|--------------------------------------|
| Description | Specifies what kind of file to emit. |
| Header File | <i>PDPrint.h</i> |
| Related Types | PDPrintParams |
| Related Methods | None |

| |
|--|
| <i>PDOutput_PS</i> Emit a PostScript file. |
| <i>PDOutput_EPSNoPrev</i> Emit an EPS file with no preview. |
| <i>PDOutput_EPSMacStdPrev</i> Emit an EPS file with standard preview. |
| <i>PDOutput_EPSMacExtPrev</i> Emit an EPS file with extended preview. |

PDPageDrawFlags

```
typedef ASUns32 PDPageDrawFlags;  
enum {  
    kPDPageDoLazyErase,  
    kPDPageUseAnnotFaces,  
    kPDPageIsPrinting  
};
```

Description Bit flags indicating how a page is rendered.

Header File *PXExpT.h*

Related Methods [PDPageDrawContentsPlaced](#)

| Flag | Description |
|-----------------------------|--|
| <i>kPDPageDoLazyErase</i> | Erase the page while rendering only as needed. |
| <i>kPDPageUseAnnotFaces</i> | Draw annotation appearances. |
| <i>kPDPageIsPrinting</i> | The page is being printed. |

PDPageMode

Description Enumerated data type. Specifies whether or not thumbnail images or bookmarks are shown.

Header File *PDExpT.h*

Related Methods [AVDocGetViewMode](#)
[AVDocSetViewMode](#)
[PDDocGetPageLabel](#)
[PDDocSetPageLabel](#)

| Value | Description |
|-----------------------|---|
| <i>PDDontCare</i> | Leaves view mode as is. |
| <i>PDUseNone</i> | Displays document, but neither thumbnails nor bookmarks. |
| <i>PDUseThumbs</i> | Displays document plus thumbnails. |
| <i>PDUseBookmarks</i> | Displays document plus bookmarks. |
| <i>PDFullScreen</i> | Displays document in full-screen viewing mode. This is equivalent to AVAppBeginFullScreen . |

PDPageRange

```
typedef struct _t_PDTextSelectRange {  
    ASInt32 startPage;  
    ASInt32 endPage;  
    ASInt32 pageSpec;  
} PDPageRange;
```

| | |
|------------------------|--|
| Description | Specifies a range of pages in a document. Page numbers begin with 0. |
| Header File | <i>PDExpt.h</i> |
| Related Types | PDPrintParams |
| Related Methods | None |

startPage

Starting page number.

endPage

Ending page number.

pageSpec

Pages in the range to print. Must be one of: *PDAllPages*, *PDEvenPagesOnly*, or *PDOddPagesOnly*. See [Page Specification](#).

PDPageStmToken

```
typedef struct _t_PageStmToken {
    ASSize_t size;
    CosType type;
    ASUns32 flags;
    ASInt32 iVal;
    char sVal[kPDPageStmStringMax];
    ASSize_t sValLen;
} PDPageStmTokenRec, *PDPageStmToken;
```

| | |
|------------------------|---|
| Description | Data structure used by PDPageStmGetToken . It contains information about the current page contents token. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPageStmGetToken |

| | |
|----------------|---|
| <i>size</i> | Size of the data structure. Must be set to <i>sizeof(PDPageStmTokenRec)</i> . |
| <i>type</i> | Token's type (<i>CosInteger</i> , <i>CosString</i> , <i>CosBoolean</i> , and so on). Must be one of the Cos Object Types . |
| <i>flags</i> | Additional information about token. Must be (see <i>PDExpT.h</i>): <i>kPDPageStmTokenHexString</i> — The token is a hexadecimal encoded string. |
| <i>iVal</i> | Token's value if the token is a Cos integer, fixed number, or name. |
| <i>sVal</i> | Token's value if the token is a Cos string. |
| <i>sValLen</i> | Length of <i>sVal</i> , in bytes. |

PDPathEnumMonitor

```
typedef struct _t_PDPathEnumMonitor {  
    ASSize_t size;  
    PDPathMoveToProc MoveTo;  
    PDPathLineToProc LineTo;  
    PDPathCurveToProc CurveTo;  
    PDPathVCurveToProc VCurveTo;  
    PDPathYCurveToProc YCurveTo;  
    PDPathRectProc Rect;  
    PDPathClosePathProc ClosePath;  
} PDPathEnumMonitorRec, *PDPathEnumMonitor;
```

| | |
|------------------------|---|
| Description | Data structure containing callbacks used by PDPathEnum . One callback is called for each path operator encountered; the callback to call depends on the operator. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDPathEnum |

size

Size of the data structure. Must be set to *sizeof(PDPathEnumMonitorRec)*.

PDPathPaintOp

Description Enumerated data type. Specifies the path painting operator used on a path.

Header File *PDExpT.h*

Related Methods [PDPathGetPaintOp](#)

| Flag | Description |
|----------------------|-------------------------------------|
| <i>pdPathNoPaint</i> | Path is not painted. |
| <i>pdPathOpClose</i> | Path contains a closepath operator. |
| <i>pdPathStroke</i> | Path contains a stroke operator. |
| <i>pdPathFill</i> | Path contains a fill operator. |
| <i>pdPathEOFill</i> | Path contains an eofill operator. |
| <i>pdPathClip</i> | Path contains a clip operator. |
| <i>pdPathEOClip</i> | Path contains an eoclip operator. |

PDPerms

| | |
|------------------------|---|
| Description | Permissions that a user has on a file. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | AVCryptGetPassword PDDocAuthorize PDDocGetPermissions |

| |
|---|
| <i>pdPermOpen</i> |
| The user is permitted to open and decrypt the document. |
| <i>pdPermSecure</i> |
| The user is permitted to change the document's security settings. |
| <i>pdPermPrint</i> |
| The user is permitted to print the document. Page Setup access is unaffected by this permission, since that affects Acrobat's preferences—not the document's. |
| <i>pdPermEdit</i> |
| The user is permitted to edit the document more than adding or modifying text notes (see also <i>pdPermEditNotes</i>). |
| <i>pdPermCopy</i> |
| The user is permitted to copy information from the document to the clipboard. |
| <i>pdPermEditNotes</i> |
| The user is permitted to add, modify, and delete text notes (see also <i>pdPermEdit</i>). |
| <i>pdPermSaveAs</i> |
| The user is permitted to perform a "Save As..." If both <i>pdPermEdit</i> and <i>pdPermEditNotes</i> are disallowed, "Save" will be disabled but "Save As..." is enabled. The "Save As..." menu item is not necessarily disabled even if the user is not permitted to perform a "Save As...". |
| <i>pdPermOwner</i> |
| The user is permitted to perform all operations, regardless of the permissions specified by the document. Unless this permission is set, the document's permissions will be reset to those in the document after a full save. |

pdPermSettable

The OR of all operations that can be set by the user in the security dialog (*pdPermPrint* + *pdPermEdit* + *pdPermCopy* + *pdPermEditNotes*).

pdPermAll

The OR of all security flags.

pdPermUser

A convenience value, that is, (*pdPermAll* – *pdPermOpen* – *pdPermSecure*).

PDPrintClient

```
typedef struct _t_PDPrintClient {
    PDDoc pdDoc;
    PDPrintParams params;

/* CONTROL STRUCTURE FOR PDDocPrintPages */

/* The following methods are listed in the order in which they
   are called. */

ACCBPROTO1 void (ACCBPROTO2* DocBegin)
(const PDPageRange ranges[], ASInt32 numRanges, ASStm stm,
const PResTree docTree, PDPrintClient printClient);

ACCBPROTO1 ASBool (ACCBPROTO2* DocSetup)
(const PDPageRange ranges[], ASInt32 numRanges,
ASStm prologStm, PDPrintClient printClient);

ACCBPROTO1 void (ACCBPROTO2* EndSetup) (ASStm stm,
PDPrintClient client);

ACCBPROTO1 void (ACCBPROTO2* NotifyNewPage) (ASInt32 pageNum,
PDPrintClient printClient);

ACCBPROTO1 void (ACCBPROTO2* PageBegin) (ASStm stm,
ASInt32 pageNum, const PResTree pageTree,
PDPrintClient printClient);

ACCBPROTO1 ASBool (ACCBPROTO2* PageSetup) (ASStm stm,
ASInt32 pageNum, PDPrintClient printClient);

ACCBPROTO1 void (ACCBPROTO2* EmitPageContents) (ASStm stm,
ASInt32 pageNum, PDPage pdPage, PDPrintClient printClient);

ACCBPROTO1 void (ACCBPROTO2* PageEnd) (ASStm stm,
ASInt32 pageNum, ASBool veryLastPage,
PDPrintClient printClient);

ACCBPROTO1 void (ACCBPROTO2* DocEnd) (ASStm stm,
```

```
PDPrintClient printClient);

/* END OF CONTROL STRUCTURE */

PDPrintGetFontEncodingMethodProc GetFontEncodingMethod;

ACCBPROTO1 ASInt32 (ACCBPROTO2* GetFontVMUsage)
(PDFont fontP, PDPrintClient printClient);

/* These routines are called before and after a font is emitted
   (outside DSC comments). */

ACCBPROTO1 void (ACCBPROTO2* EmitPSFontBegin) (ASStm stm,
PDFont fontP, PDPrintClient printClient);

ACCBPROTO1 void (ACCBPROTO2* EmitPSFontEnd) (ASStm stm,
PDFont fontP, PDPrintClient printClient);

/* These routines are called before and after a font encoding is
   emitted. */

ACCBPROTO1 void (ACCBPROTO2* EmitPSFontEncodingBegin)
(ASStm stm, PDFont fontP, PDPrintClient printClient);

ACCBPROTO1 void (ACCBPROTO2* EmitPSFontEncodingEnd)
(ASStm stm, PDFont fontP, PDPrintClient printClient);

/* These 2 functions are called before and after a PostScript
   resource is emitted. */

ACCBPROTO1 void (ACCBPROTO2* EmitPSResourceBegin) (ASStm stm,
ASAtom resType, char* resName, CosObj resObj,
PDPrintClient printClient);

ACCBPROTO1 void (ACCBPROTO2* EmitPSResourceEnd) (ASStm stm,
ASAtom resType, char* resName, CosObj resObj,
PDPrintClient printClient);

ACCBPROTO1 ASBool (ACCBPROTO2* DoCancel)
```

```
(PDPrintClient printClient);

ACCBPROTO1 ASInt32 (ACCBPROTO2* FlushString) (char* data,
ASInt32 nData, PDPrintClient printClient);

/* These 2 routines are used to create the PostScript prolog. */

ACCBPROTO1 ASInt32 (ACCBPROTO2* EmitPrologString)
(const char* s, ASInt32 nData, PDPrintClient printClient);

PDPrintEmitPrologResourceProc EmitPrologResource;

PDPrintCanEmitFontProc CanEmitFont;

PDPrintEmitFontProc EmitFont;

void* clientData;
} PDPrintClientRec, *PDPrintClient;
```

Description *(Placed PDF only, required for PostScript printing)* Data structure used by [PDDocPrintPages](#). Contains methods to be implemented by the client. Unless otherwise indicated, methods may be *NULL*, indicating that they do nothing.

Related Methods [PDDocPrintPages](#)

| | |
|-------------------|--|
| <i>pdDoc</i> | The document to print. |
| <i>params</i> | A control structure describing how to print the document. Must be a PDPrintParams structure. |
| <i>clientData</i> | Data for the client's use. |

PDPrintParams

```
typedef struct {
    ASSize_t size;
    PDPageRange* ranges;
    ASInt32 numRanges;
    ASBool shrinkToFit;
    ASBool emitAnnotForms;
    ASBool emitPS;
    ASInt32 psLevel;
    PDOutputType outputType;
    PDInclusion incBaseFonts;
    PDInclusion incEmbeddedFonts;
    PDInclusion incType1Fonts;
    PDInclusion incType3Fonts;
    PDInclusion incTrueTypeFonts;
    PDInclusion incCIDFonts;
    PDInclusion incProcsets;
    PDInclusion incOtherResources;
    ASInt32 fontPerDocVM;
    ASBool emitShowpage;
    ASBool emitTTFontsFirst;
    ASBool setPageSize;
    ASBool emitDSC;
    ASBool setupProcsets;
    ASBool emitColorSeps;
    ASBool binaryOK;
    ASBool emitRawData;
    ASBool TTasT42;
    float scale;
    ASBool emitExternalStreamRef;
    ASBool emitHalftones;
    ASBool centerCropBox;
    ASBool useFontAliasNames;
    ASBool emitPageRotation;
    ASBool emitSeparableImagesOnly;
    ASBool emitExtGState;
    ASFixedRect boundingBox;
} PDPrintParamsRec, *PDPrintParams;
```

| | |
|------------------------|---|
| Description | Data structure indicating how a document should be printed. |
| Header File | <i>PDPrint.h</i> |
| Related Methods | PDFLPrintDoc |

| | |
|-------------------------|--|
| <i>size</i> | Size of the data structure. Must be set to <i>sizeof(PDPrintParams)</i> . |
| <i>ranges</i> | Ranges of pages to print. Use <i>NULL</i> to print the entire document. |
| <i>numRanges</i> | Number of ranges of pages to print in <i>ranges</i> . Default is 0. |
| <i>shrinkToFit</i> | <i>true</i> if the page is scaled to fit the printer page size, <i>false</i> otherwise. This field overrides <i>scale</i> . Default is <i>false</i> . |
| <i>emitAnnotForms</i> | Emit annotations that contain a Form <i>XObject</i> . Default is <i>true</i> . |
| <i>emitPS</i> | If <i>true</i> , emit a PostScript file. Default is <i>true</i> . |
| <i>psLevel</i> | PostScript level: 1 or 2. Default is 2. |
| <i>outputType</i> | Print PostScript or EPS with or without a preview. <i>Note: For fonts, the following 6 parameters are applied in the order in which they are appear in this table. For example, an embedded Type 1 font follows the rule for embedded fonts, not the rule for Type 1 fonts.</i> |
| <i>incBaseFonts</i> | Embed the base fonts. Default is <i>kIncludeNever</i> . |
| <i>incEmbeddedFonts</i> | Embed fonts that are embedded in the PDF file. This overrides the <i>incType1Fonts</i> , <i>incTrueTypeFonts</i> , and <i>incCIDFonts</i> fields. Default is <i>kIncludeOncePerDoc</i> . |
| <i>incType1Fonts</i> | Embed Type 1 fonts. Default is <i>kIncludeOncePerDoc</i> . |
| <i>incType3Fonts</i> | Embed Type 3 fonts. Default is <i>kIncludeOnEveryPage</i> . <i>Note: This parameter must always be set to kIncludeOnEveryPage. PDF files exist with Type 3 fonts that contain different encodings on different pages.</i> |

| |
|---|
| <i>incTrueTypeFonts</i> |
| Embed TrueType fonts. Default is <i>kIncludeOncePerDoc</i> . |
| <i>incCIDFonts</i> |
| Embed CID fonts. Default is <i>kIncludeOncePerDoc</i> . |
| <i>incProcsets</i> |
| Include Procsets in the file. Default is <i>kIncludeOncePerDoc</i> . |
| <i>incOtherResources</i> |
| Include all other types of resources in the file. Default is <i>kIncludeOncePerDoc</i> . |
| <i>fontPerDocVM</i> |
| Amount of VM available for font downloading at the document level. Ignored if ≤ 0 . |
| <i>Note: This must be set to 0 for the toolkit; it is only used by the viewer.</i> |
| <i>emitShowpage</i> |
| Emit save and restore showpage in PostScript files. Default is <i>true</i> . |
| <i>emitTTFontsFirst</i> |
| Emit TrueType fonts before <i>any</i> other fonts. Default is <i>false</i> . |
| <i>setPageSize</i> |
| (PostScript level 2 only) Set the page size on each page. Use the media box for outputting to PostScript files, use the crop box for EPS files. Default is <i>false</i> . |
| <i>emitDSC</i> |
| Write DSC (Document Structuring Conventions) comments. Default is <i>true</i> . |
| <i>setupProcsets</i> |
| If procsets are included, also include init/term code. Default is <i>true</i> . |
| <i>Note: This must be set to true for the toolkit.</i> |
| <i>emitColorSeps</i> |
| Emit images for Level-1 separations. Default is <i>false</i> . |
| <i>binaryOK</i> |
| <i>true</i> if binary data is permitted in the PostScript file, <i>false</i> otherwise. Default is <i>true</i> . |

| |
|--|
| <i>emitRawData</i> <i>true</i> if add no unnecessary filters when emitting image data, <i>false</i> otherwise. Default is <i>false</i> . |
| <i>TTasT42</i> If including TrueType fonts, convert to Type 42 fonts instead of Type 1 fonts. Default is <i>false</i> . |
| <i>scale</i> Document-wide scale factor. 100.0 = full size. Default is 100. |
| <i>emitExternalStreamRef</i> If an Image resource uses an external stream, emit code that points to the external file. Default is <i>false</i> . <i>Note: This must be set to false.</i> |
| <i>emitHalftones</i> Preserve any halftone screening in the PDF file. Default is <i>false</i> . |
| <i>centerCropBox</i> <i>true</i> if CropBox output is centered on the page when the CropBox < MediaBox, <i>false</i> otherwise. Default is <i>true</i> . |
| <i>useFontAliasNames</i> Used when printing with system fonts. Default is <i>false</i> . |
| <i>emitPageRotation</i> Emit a concat at the beginning of each page so that the page is properly rotated. Used when emitting EPS. Default is <i>false</i> . |
| <i>emitSeparableImagesOnly</i> If emitting EPS, include only CMYK and gray images. |
| <i>emitExtGState</i> Omit all extended graphic state parameters. This overrides <i>emitHalftones</i> . If <i>false</i> , halftones are not emitted. Default is <i>true</i> . |
| <i>boundingBox</i> If all zeroes, is ignored. Otherwise, is used for <i>%%BoundingBox</i> DSC comment and in <i>centerCropBox</i> calculations and for <i>setpagedevice</i> . Default is [0 0 0 0]. |

PDResourceEnumMonitor

```
typedef struct _t_PDResourceEnumMonitor {  
    ASSize_t size;  
    PDResourceEnumFontProc EnumFont;  
    PDResourceEnumXObjectProc EnumXObject;  
    PDResourceEnumProcSetProc EnumProcSet;  
    PDResourceEnumColorSpaceProc EnumColorSpace;  
} PDResourceEnumMonitorRec, *PDResourceEnumMonitor;
```

| | |
|------------------------|---|
| Description | Data structure containing callbacks used when enumerating the resources of a form with PDFFormEnumResources . |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDFFormEnumResources |

size

Size of the data structure. Must be set to *sizeof(PDResourceEnumMonitorRec)*.

PDRotate

```
typedef AEnum16 PDRotate;
```

Description Enumerated data type. Specifies page rotation, in degrees.

Header File *PDExpT.h*

Related Methods [PDPageGetRotate](#)
[PDPageSetRotate](#)

| |
|--------------------|
| <i>pdRotate0</i> |
| <i>pdRotate90</i> |
| <i>pdRotate180</i> |
| <i>pdRotate270</i> |

PDSaveFlags

| | |
|------------------------|--|
| Description | Enumerated data type. Specifies options for saving a file. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDDocSave |

PDSaveFull

Write the entire file to the filename specified by *newPath*. In addition to this flag, the *PDSaveCollectGarbage*, *PDSaveCopy*, and *PDSaveLinearized* may be specified. Plug-ins that use *PDSaveFull* are also encouraged to use *PDSaveCollectGarbage*.

PDSaveCollectGarbage

Remove unreferenced objects, often reducing file size. Plug-ins are encouraged to use this flag. This flag can only be specified if *PDSaveFull* is also used.

PDSaveCopy

Write a copy of the file into the file specified by *newPath*, but keep using the old file. This flag can only be specified if *PDSaveFull* is also used.

PDSaveLinearized

Write the file linearized for page-served remote (network) access. This flag can only be specified if *PDSaveFull* is also used.

During linearization, all Cos objects in the PDF file can be (and usually are) renumbered and recached in memory. Any PD or Cos level objects stored prior to linearization are *invalidated* if *PDSaveLinearized* is set. If invalid objects are used as a parameter in any method, a [genErrBadParm](#) (bad parameter) exception is raised.

Thus, if any objects have been acquired, they need to be released prior to saving a PDDoc linearized. Register for the [PDDocWillSave](#) and [PDDocDidSave](#) notifications if the plug-in has acquired any objects; release them after the [PDDocWillSave](#) notification and acquire them again after the [PDDocDidSave](#) notification.

PDSaveBinaryOK

It is OK to store binary data in the PDF file.

PDSysFontMatchFlags

```
typedef enum {  
    kPDSysFontMatchNameAndCharSet,  
    kPDSysFontMatchFontType,  
    kPDSysFontMatchWritingMode  
} PDSysFontMatchFlags;
```

Description Font matching flags for [PDFindSysFontForPDEFont](#) and [PDFindSysFont](#).

Header File *PSFExpT.h*

Related Methods [PDFindSysFont](#)
[PDFindSysFontForPDEFont](#)

| |
|--|
| <i>kPDSysFontMatchNameAndCharSet</i> |
| Match the font name and character set. |
| <i>kPDSysFontMatchFontType</i> |
| Match the font type. |
| <i>kPDSysFontMatchWritingMode</i> |
| Match the writing mode, that is, horizontal or vertical. |

PDTextSelectRange

```
typedef struct _t_PDTextSelectRange {  
    ASInt32 start;  
    ASInt32 end;  
    ASInt32 ofsStart;  
    ASInt32 ofsEnd;  
} PDTextSelectRangeRec, *PDTextSelectRange;
```

Description

Data structure used to specify a range of text in a text selection.

Use 0 for *ofsStart* and *ofsEnd* for whole-word selections. Nonzero values for *ofsStart* and *ofsEnd* are supported by *PDText* but are currently ignored by the Acrobat viewer's user interface code (which highlights only whole-word selections). If *ofsEnd* is 0, *end* is the first word *not* selected.

Header File

PDExpT.h

Related Methods

[PDTextSelectCreateRanges](#)
[PDTextSelectGetRange](#)

start

The word containing start of selection.

end

The word containing end of selection.

ofsStart

The offset into word at start of selection.

ofsEnd

The offset into word at end of selection.

PDTextState

PDTextStateP

```
typedef struct _t_PDTextState {  
    PDFont font;  
    ASFixed charSpacing;  
    ASFixed wordSpacing;  
    ASFixed horizontalScale;  
    ASFixed leading;  
    ASFixed textRise;  
    ASFixed textSize;  
    ASInt32 renderMode;  
    ASFixedMatrix textMatrix;  
} PDTextState, *PDTextStateP;
```

| | |
|------------------------|---|
| Description | Data structure containing information about the current text state. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDTextGetState |

PDThumbCreationServer

```
typedef struct _t_PDThumbCreationServer {  
    ASSize_t size;  
    PDThumbCreationNotifyPageProc NotifyPage;  
    PDThumbCreationGetThumbDataProc GetThumbData;  
    PDThumbCreationDrawThumbProc DrawThumb  
} PDThumbCreationServerRec, *PDThumbCreationServer;
```

Description Data structure containing callbacks that implement a creation server. The callbacks implement the creation server functions.

Header File *PDExpt.h*

Related Methods [PDDocCreateThumbs](#)

size

Size of the data structure. Must be set to `sizeof(PDThumbCreationServerRec)`.

PIHandshakeData_V0200























```
typedef struct {
    ASUns32 handshakeVersion;
    ASAtom appName;
    ASAtom extensionName;
    PIExportHFTsProcType exportHFTsCallback;
    PIImportReplaceAndRegisterProcType
importReplaceAndRegisterCallback;
    PIInitProcType initCallback;
    PIUnloadProcType unloadCallback;
} PIHandshakeData_V0200;
```














| | |
|--------------------------|--|
| Description | Structure describing plug-in data for handshaking with the Acrobat viewer. |
| Header File | <i>PIVersn.h</i> |
| Related Callbacks | PIHandshake |
| Related Methods | None |

| |
|---|
| <i>handshakeVersion</i> |
| (Required) Handshake version. Always use <i>HANDSHAKE_V0200</i> . |
| <i>appName</i> |
| (Optional) Name of host application. |
| <i>extensionName</i> |
| (Required) Name of the plug-in. |
| <i>exportHFTsCallback</i> |
| (Optional) Callback to register the HFTs this plug-in is exporting. May be <i>NULL</i> . |
| <i>importReplaceAndRegisterCallback</i> |
| (Optional) Callback to import other plug-in's HFTs, replace HFT functions, and register for notifications. May be <i>NULL</i> . |
| <i>initCallback</i> |
| (Required) Callback for plug-in to initialize itself. |
| <i>unloadCallback</i> |
| (Optional) Callback to clean up when the plug-in terminates. May be <i>NULL</i> . |

Predefined Cursors

| | |
|------------------------|--|
| Description | A group of predefined cursors. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVSysGetStandardCursor |

| Cursor | |
|----------------------------|---|
| <i>ARROW_CURSOR</i> |  |
| <i>IBEAM_CURSOR</i> |  |
| <i>CROSSHAIR_CURSOR</i> |  |
| <i>BOX_IBEAM_CURSOR</i> |  |
| <i>HAND_CURSOR</i> |  |
| <i>FIST_CURSOR</i> |  |
| <i>ZOOM_IN_CURSOR</i> |  |
| <i>ZOOM_OUT_CURSOR</i> |  |
| <i>ZOOM_MAX_CURSOR</i> |  |
| <i>LINK_CURSOR</i> |  |
| <i>GROW_CURSOR</i> |  |
| <i>BAR_IBEAM_CURSOR</i> |  |
| <i>WAIT_CURSOR</i> |  |
| <i>MOVEPAGE_CURSOR</i> |  |
| <i>COPYPAGE_CURSOR</i> |  |
| <i>MOVEPAGES_CURSOR</i> |  |
| <i>COPYPAGES_CURSOR</i> |  |
| <i>REPLACEPAGE_CURSOR</i> |  |
| <i>REPLACEPAGES_CURSOR</i> |  |
| <i>NOP_CURSOR</i> |  |
| <i>THREAD_CURSOR</i> |  |
| <i>WORDFINDER_CURSOR</i> |  |

| Cursor | |
|----------------------------------|---|
| <i>HIDDEN_CURSOR</i> | |
| <i>GROWTOPLEFT_CURSOR</i> |  |
| <i>GROWBOTTOMLEFT_CURSOR</i> |  |
| <i>MOVE_CURSOR</i> |  |
| <i>HAND_THREAD_UP_CURSOR</i> |  |
| <i>HAND_THREAD_END_CURSOR</i> |  |
| <i>HAND_THREAD_UP_END_CURSOR</i> |  |
| <i>HAND_THREAD_BEGIN_CURSOR</i> |  |
| <i>THREAD_CONNECT_CURSOR</i> |  |
| <i>THREAD_END_CURSOR</i> |  |
| <i>VERT_IBEAM_CURSOR</i> |  |
| <i>GROWLEFTRIGHT_CURSOR</i> |  |
| <i>HIGHLIGHT_CURSOR</i> |  |
| <i>GROWTOPBOTTOM_CURSOR</i> |  |

ProgressMonitor

```
typedef struct _t_ProgressMonitor {  
    ASSize_t size;  
    PMBeginOperationProc beginOperation;  
    PMEndOperationProc endOperation;  
    PMSetDurationProc setDuration;  
    PMSetCurrValueProc setCurrValue;  
    PMGetDurationProc getDuration;  
    PMGetCurrValueProc getCurrValue;  
} ProgressMonitorRec, *ProgressMonitor;
```

Description Data structure containing callbacks that implement a progress monitor. The callbacks implement the progress monitor functions. A progress monitor is used to display progress during potentially time-consuming operations. Progress monitors are included as parameters in many API calls. The Acrobat viewer's built-in progress monitor can be obtained by calling [AVAppGetDocProgressMonitor](#).

Header File *ASExpT.h*

Related Methods [AVAppGetDocProgressMonitor](#)
[PDDocSave](#)

size

Size of the data structure. Must be set to *sizeof(ProgressMonitorRec)*.

Punctuation Characters

| | |
|------------------------|--|
| Description | Constants that specify various punctuation characters. |
| Header File | <i>WinAnsiResources.c</i> |
| Related Methods | Numerous |

| Character | Description |
|------------------------|-------------|
| <i>acute</i> | ` |
| <i>ampersand</i> | & |
| <i>asciicircumflex</i> | ^ |
| <i>asciitilde</i> | ~ |
| <i>asterisk</i> | * |
| <i>at</i> | @ |
| <i>backslash</i> | \ |
| <i>bar</i> | |
| <i>braceleft</i> | { |
| <i>braceright</i> | } |
| <i>bracketleft</i> | [|
| <i>bracketright</i> |] |
| <i>cedilla</i> | ¸ |
| <i>cent</i> | ¢ |
| <i>colon</i> | : |
| <i>comma</i> | , |
| <i>copyright</i> | © |
| <i>currency</i> | ¤ |
| <i>degree</i> | ° |
| <i>dieresis</i> | ¨ |
| <i>divide</i> | ÷ |
| <i>dollar</i> | \$ |

| Character | Description |
|--------------------------|-------------|
| <i>equal</i> | = |
| <i>exclamation point</i> | ! |
| <i>exclamdown</i> | ¡ |
| <i>grave</i> | ` |
| <i>greaterthan</i> | > |
| <i>guillemotleft</i> | « |
| <i>guillemotright</i> | » |
| <i>hyphen</i> | - |
| <i>lessthan</i> | < |
| <i>logicalnot</i> | ¬ |
| <i>macron</i> | ˉ |
| <i>multiply</i> | * |
| <i>numbersign</i> | # |
| <i>onehalf</i> | ½ |
| <i>onequarter</i> | ¼ |
| <i>ordfeminine</i> | ª |
| <i>ordmasculine</i> | º |
| <i>paragraph</i> | ¶ |
| <i>parenleft</i> | (|
| <i>parenright</i> |) |
| <i>percent</i> | % |
| <i>period</i> | . |
| <i>periodcentered</i> | · |
| <i>plus</i> | + |
| <i>plusminus</i> | ± |
| <i>question mark</i> | ? |
| <i>questiondown</i> | ¿ |
| <i>quotedbl</i> | " |

| Character | Description |
|----------------------|---------------|
| <i>quotesingle</i> | ' |
| <i>registered</i> | ® |
| <i>section</i> | § |
| <i>semicolon</i> | ; |
| <i>slash</i> | / |
| <i>space</i> | " " |
| <i>sterling</i> | £ |
| <i>threequarters</i> | $\frac{3}{4}$ |
| <i>underscore</i> | — |
| <i>yen</i> | ¥ |

Security Info Flags

| | |
|------------------------|--|
| Description | Constants used to specify various information about the Acrobat viewer's security and permissions. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDDocGetNewSecurityInfo |

| |
|---|
| <i>pdInfoHasOwnerPW</i> The document has an owner password. |
| <i>pdInfoHasUserPW</i> The document has a user password. |
| <i>pdInfoCanPrint</i> The document can be printed. |
| <i>pdInfoCanEdit</i> The document can be modified, for example by adding notes, links, or bookmarks (see also <i>pdInfoCanEditNotes</i>). |
| <i>pdInfoCanCopy</i> The document text and graphics can be copied to the clipboard. |
| <i>pdInfoCanEditNotes</i> The document's notes, but nothing else, can be modified (see also <i>pdInfoCanEdit</i>). |

StdPassword

```
typedef char StdPassword[MAX_PWCHARS+1];
```

| | |
|--------------------------|--|
| Description | Character string containing a password for the standard Acrobat viewer security handler. |
| Header File | <i>PICrypt.h</i> |
| Related Types | StdSecurityData |
| Related Callbacks | None |
| Related Methods | None |

StdSecurityData

```
typedef struct _t_StdSecurityData {
    os_size_t size;
    ASBool newUserPW;
    ASBool hasUserPW;
    StdPassword userPW;
    ASBool newOwnerPW;
    ASBool hasOwnerPW;
    StdPassword ownerPW;
    PDPerms perms;
} StdSecurityDataRec, *StdSecurityData;
```

| | |
|--------------------------|---|
| Description | Structure describing the data for the standard security handler provided in the Acrobat viewer. |
| Header File | <i>PICrypt.h</i> |
| Related Callbacks | None |
| Related Methods | None |

| | |
|-------------------|--|
| <i>size</i> | Size of the data structure. Must be set to <i>sizeof(StdSecurityDataRec)</i> . |
| <i>newUserPW</i> | <i>true</i> if the user password should be changed, <i>false</i> otherwise. |
| <i>hasUserPW</i> | <i>true</i> if a user password is provided, <i>false</i> otherwise. |
| <i>userPW</i> | The user password. |
| <i>newOwnerPW</i> | <i>true</i> if the owner password should be changed, <i>false</i> otherwise. |
| <i>hasOwnerPW</i> | <i>true</i> if an owner password is provided, <i>false</i> otherwise. |
| <i>ownerPW</i> | The owner password. |
| <i>perms</i> | The permissions to allow for a file. See <i>PDTypes.h</i> . |

Tool Button Flags

| | |
|------------------------|--|
| Description | Constants that specify whether a tool button is external to the viewer or not. |
| Header File | <i>AVExpT.h</i> |
| Related Methods | AVToolButtonSetExternal |

| |
|---|
| <i>TOOLBUTTON_INTERNAL</i> |
| Indicates tool button is visible only in the viewer's toolbar. |
| <i>TOOLBUTTON_EXTERNAL</i> |
| Indicates tool button is visible only in the tool bar of an external application (such as a Web browser). |

Transition Duration

| | |
|------------------------|---|
| Description | Duration of a PDTrans . |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDTransNew |

| |
|---|
| <i>fxDefaultPageDuration</i> |
| Constant used to indicate that there is no page timing information available. |
| <i>fxDefaultTransDuration</i> |
| Default duration for a transition. |

TKAllocatorProcs

TKAllocatorProcsP

```
typedef struct _t_TKAllocatorProcs {
    ASMemAllocProc allocProc;
    ASMemReallocProc reallocProc;
    ASMemFreeProc freeProc;
    ASMemAvailProc memAvailProc;
    void* clientData;
} TKAllocatorProcs, *TKAllocatorProcsP;
```

| | |
|--------------------------|---|
| Description | Structure for memory allocation callbacks used by the PDF Library. |
| Header File | <i>PDFInit.h</i> |
| Related Callbacks | ASMemAllocProc ASMemAvailProc ASMemFreeProc ASMemReallocProc |
| Related Methods | PDFLInit |

allocProc

[ASMemAllocProc](#) callback — Called when the PDF Library needs to allocate memory.

reallocProc

[ASMemReallocProc](#) callback — Called when the PDF Library needs to reallocate a block of memory.

freeProc

[ASMemFreeProc](#) callback — Called when the PDF Library needs to free a block of memory.

memAvailProc

[ASMemAvailProc](#) callback — Called when the PDF Library needs to determine the amount of memory available.

clientData

A void* of data to pass into each of the memory management callbacks.

TKResourceProcs

TKResourceProcsP

```
typedef struct _t_TKResourceProcs {  
    TKResourceAcquireProc acquireProc;  
    TKResourceReleaseProc releaseProc;  
    void* clientData;  
} TKResourceProcs, *TKResourceProcsP;
```

| | |
|--------------------------|--|
| Description | Structure for resource acquisition callbacks used by the PDF Library. |
| Header File | <i>PDFInit.h</i> |
| Related Callbacks | TKResourceAcquireProc TKResourceReleaseProc |
| Related Methods | PDFLInit |

acquireProc

[TKResourceAcquireProc](#) callback — Called when the PDF Library needs to acquire resources.

releaseProc

[TKResourceReleaseProc](#) callback — Called when the PDF Library needs to release resources.

clientData

A void* of data to pass into every call to the [TKResourceAcquireProc](#) or [TKResourceReleaseProc](#) as the *clientData* argument.

WinPort

```
typedef struct _t_WinPort {  
    HWND hWnd;  
    HDC hDC;  
} WinPortRec, *WinPort;
```

| | |
|------------------------|--|
| Description | The <i>HWND</i> is that of the document window's <i>AVPageView</i> region (the portion of the window in which the PDF file's pages are drawn). |
| Header File | <i>AVExpT.h</i> |
| Related Methods | <u>AVPageViewAcquireMachinePort</u> |

Word Attributes

| | |
|------------------------|---|
| Description | Constants that specify various attributes of words. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDWordGetAttr |

WXE_ADJACENT_TO_SPACE

The character following the end of the word is a space (either an explicit space character encoded in a string, or one that appears implicitly because the drawing point was moved).

WXE_HAS_UNMAPPED_CHAR

One or more characters in the word cannot be represented in the output font encoding.

WXE_HAS_LIGATURE

The word contains a ligature.

WXE_HAS_DIGIT

One or more characters in the word are digits.

WXE_HAS_HYPHEN

There is a hyphen in the word.

WXE_HAS_SOFT_HYPHEN

There is a soft hyphen in the word.

WXE_HAS_PUNCTUATION

One or more characters in the word are punctuation marks. Other flag bits can be checked to test whether the punctuation was at the beginning of the word (*WXE_HAS_LEADING_PUNC*), the end of the word (*WXE_HAS_TRAILING_PUNC*), or elsewhere in the word.

WXE_HAS_LEADING_PUNC

The first character in the word is a punctuation mark. If this bit is set, *WXE_HAS_PUNCTUATION* will also be set.

WXE_HAS_TRAILING_PUNC

The last character in the word is a punctuation mark. If this bit is set, *WXE_HAS_PUNCTUATION* will also be set.

WXE_HAS_NONALPHANUM

The word contains a character outside the range of A-Z, a-z, 0-9.

WXE_HAS_LETTER

The word contains a character between A-Z or a-z.

WXE_HAS_UPPERCASE

The word contains a character between A-Z.

Word Finder Sort Order Flags

| | |
|------------------------|---|
| Description | Constants that specify which tables the word finder is to fill. |
| Header File | <i>PDExpT.h</i> |
| Related Methods | PDDocCreateWordFinder PDDocCreateWordFinderUCS |

WXE_PDF_ORDER

Enumerates words in the order they appear in the PDF file. This order may not be the same as that in which a person would read them on a page.

WXE_XY_SORT

Enumerates words sorted by their *x*- and *y*-coordinates on the page. For a page with a single column of text, this is usually close to the order in which a person would read the text on the page.

Lists

ASGetConfiguration Selectors

CanEdit — *ASBool*

Whether or not editing is allowed. Checking this is the correct way to determine whether one is in an editing environment (for example, Acrobat) as opposed to a non-editing environment (for example, Reader).

Product — *const char **

Acrobat — "Acrobat"

Reader — "Reader"

Acrobat Toolkit — "Acrobat Toolkit" (note that this name contains a space)

Exchange LE (does not exist in version 2.1 and later) — "Exchange LE"
(note that this name contains a space)

The quotes are shown for clarity only; they are not part of the name that is returned. Do not rely on the product name to determine whether or not the product can edit files, use the *CanEdit* selector to do this.

CosObjEnum Actions

Methods: [CosObjEnum](#)

| <i>Object type</i> | <i>Action</i> |
|-------------------------|--|
| <i>scalar or string</i> | Returns <i>true</i> . |
| <i>dictionary</i> | Calls <i>proc</i> for each key/value pair. <i>obj</i> is key, <i>value</i> is value. |
| <i>array</i> | Calls <i>proc</i> for each element. <i>obj</i> is element, <i>value</i> is invalid. |
| <i>stream</i> | Calls <i>proc</i> on stream's attribute dictionary. <i>obj</i> is dict, <i>value</i> is invalid. |

Enumerators

| |
|--|
| <u>ASEnumExtensions</u> |
| <u>AVAppEnumActionHandlers</u> |
| <u>AVAppEnumAnnotHandlers</u> |
| <u>AVAppEnumDocs</u> |
| <u>AVAppEnumSystemFonts</u> |
| <u>AVAppEnumTools</u> |
| <u>AVAppEnumTransHandlers</u> |
| <u>AVDocEnumSelection</u> |
| <u>AVDocSelectionEnumPageRanges</u> |
| <u>AVMenubarAcquireMenuByPredicate</u> |
| <u>AVMenubarAcquireMenuItemByPredicate</u> |
| <u>AVToolBarEnumButtons</u> |
| <u>CosDocEnumEOFs</u> |
| <u>CosDocEnumIndirect</u> |
| <u>CosObjEnum</u> |
| <u>PDCharProcEnum</u> |
| <u>PDDocEnumFonts</u> |
| <u>PDDocEnumLoadedFonts</u> |
| <u>PDDocEnumResources</u> |
| <u>PDEAttrEnumTable</u> |
| <u>PDEEnumElements</u> |
| <u>PDEClipFlattenedEnumElems</u> |
| <u>PDEClipFlattenedEnumElems</u> |
| <u>PDEnumDocs</u> |
| <u>PDEnumSysFonts</u> |
| <u>PDEObjectDump</u> |
| <u>PDFontEnumCharProcs</u> |

| |
|--|
| <u>PDFormEnumPaintProc</u> |
| <u>PDFormEnumResources</u> |
| <u>PDNameTreeEnum</u> |
| <u>PDNumTreeEnum</u> |
| <u>PDPageEnumContents</u> |
| <u>PDPageEnumResources</u> |
| <u>PDPathEnum</u> |
| <u>PDTextEnum</u> |
| <u>PDTextSelectEnumQuads</u> |
| <u>PDTextSelectEnumText</u> |
| <u>PDWordFinderEnumWords</u> |
| <u>PDXObjectEnumFilters</u> |
| <u>PDXObjectGetData</u> |

Font Subtypes

Methods: [PDFontGetSubtype](#)

| <i>Subtype</i> | <i>Description</i> |
|---------------------|--|
| <i>CIDFontType0</i> | Type 0 CID font |
| <i>CIDFontType2</i> | Type 2 CID font |
| <i>Type0</i> | Type 0 PostScript font |
| <i>Type1</i> | Type 1 PostScript font |
| <i>Type3</i> | Type 3 PostScript font |
| <i>MMType1</i> | Type 1 multiple master PostScript font |
| <i>TrueType</i> | TrueType font |

Glyph Names of Word Separators

Methods: [PDDocCreateWordFinder](#)
[PDDocCreateWordFinderUCS](#)
[PDWordSplitString](#)

| | | | | |
|--------------|------------|----------------|----------------|---------------|
| ampersand | comma | greater | parenleft | registered |
| asciicircum | copyright | guillemotleft | parenright | section |
| asciitilde | currency | guillemotright | percent | semicolon |
| asterisk | dagger | guilsinglleft | period | slash |
| at | daggerdbl | guilsinglright | periodcentered | space |
| backslash | degree | hyphen | perthousand | sterling |
| bar | divide | less | plus | threequarters |
| braceleft | dollar | logicalnot | plusminus | threesuperior |
| braceright | ellipsis | multiply | question | tilde |
| bracketleft | emdash | numbersign | questiondown | trademark |
| bracketright | endash | onehalf | quotedbl | twosuperior |
| brokenbar | equal | onequarter | quotedblbase | underscore |
| bullet | exclam | onesuperior | quotedblleft | yen |
| cent | exclamdown | ordfeminine | quotedblright | |
| circumflex | florin | ordmasculine | quoteleft | |
| colon | fraction | paragraph | quotesinglbase | |

Key Codes

Header file: *ASKey.h*

| <i>Key Code—All platforms</i> | Value |
|-------------------------------|-------|
| <i>ASKEY_ARROW_D</i> | 31 |
| <i>ASKEY_ARROW_L</i> | 28 |
| <i>ASKEY_ARROW_R</i> | 29 |
| <i>ASKEY_ARROW_U</i> | 30 |
| <i>ASKEY_ESCAPE</i> | 27 |
| <i>ASKEY_HELP</i> | 5 |
| <i>ASKEY_PAGE_D</i> | 12 |
| <i>ASKEY_PAGE_U</i> | 11 |
| <i>ASKEY_SPACE</i> | 32 |
| <i>ASKEY_TAB</i> | 9 |

| <i>Key Code—Windows</i> | Value |
|-------------------------|-------|
| <i>ASKEY_DEL</i> | 127 |
| <i>ASKEY_END</i> | 1 |
| <i>ASKEY_ENTER</i> | 13 |
| <i>ASKEY_HOME</i> | 4 |
| <i>ASKEY_MENU</i> | 2 |

| <i>Key Code—Macintosh</i> | Value |
|---------------------------|-------|
| <i>ASKEY_CLEAR</i> | 27 |
| <i>ASKEY_CR</i> | 13 |
| <i>ASKEY_DEL</i> | 8 |
| <i>ASKEY_END</i> | 4 |
| <i>ASKEY_ENTER</i> | 3 |
| <i>ASKEY_HOME</i> | 1 |

| <i>Key Code—UNIX</i> | Value |
|----------------------|-------|
| <i>ASKEY_CLEAR</i> | 27 |
| <i>ASKEY_CR</i> | 13 |
| <i>ASKEY_DEL</i> | 8 |
| <i>ASKEY_END</i> | 4 |
| <i>ASKEY_ENTER</i> | 10 |
| <i>ASKEY_HOME</i> | 1 |

Language Codes

Methods: [AVAppGetLanguage](#)

| <i>Code</i> | <i>Language</i> |
|-------------|-----------------|
| <i>DEU</i> | German |
| <i>ENU</i> | English |
| <i>ESP</i> | Spanish |
| <i>FRA</i> | French |
| <i>ITA</i> | Italian |
| <i>NLD</i> | Dutch |
| <i>SVE</i> | Swedish |

Menu Item Names

Apple menu item names (Macintosh)

Methods: [AVMenubarAcquireMenuItemByName](#)

| |
|---|
| About |
| AboutExtensions |
| endAboutGroup <i>(New in Acrobat 2.0)</i> |

Document Menu Item Names

Methods: [AVMenubarAcquireMenuItemByName](#)

| |
|--|
| FirstPage <i>(Moved from View menu in Acrobat 4.0)</i> |
| PrevPage <i>(Moved from View menu in Acrobat 4.0)</i> |
| NextPage <i>(Moved from View menu in Acrobat 4.0)</i> |
| LastPage <i>(Moved from View menu in Acrobat 4.0)</i> |
| GoToPage <i>(Moved from View menu in Acrobat 4.0)</i> |
| endPageNavGroup <i>(Moved from View menu in Acrobat 4.0)</i> |
| GoBackDoc <i>(New in Acrobat 4.0)</i> |
| GoBack <i>(Moved from View menu in Acrobat 4.0)</i> |
| GoForward <i>(Moved from View menu in Acrobat 4.0)</i> |
| GoForwardDoc <i>(New in Acrobat 4.0)</i> |
| endGoBackGroup <i>(Moved from View menu in Acrobat 4.0)</i> |
| InsertPages <i>(Not in Reader, moved from Edit menu in Acrobat 4.0)</i> |
| ExtractPages <i>(Not in Reader, moved from Edit menu in Acrobat 4.0)</i> |
| ReplacePages <i>(Not in Reader, moved from Edit menu in Acrobat 4.0)</i> |
| DeletePages <i>(Not in Reader, moved from Edit menu in Acrobat 4.0)</i> |
| endDocumentOpGroup <i>(New in Acrobat 3.0)</i> |
| CropPages <i>(Not in Reader, moved from Edit menu in 3.0)</i> |
| RotatePages <i>(Not in Reader, moved from Edit menu in 3.0)</i> |
| endPageOpGroup <i>(Not in Reader, moved from Edit menu in 3.0)</i> |
| NumberPages <i>(Not in Reader, new in Acrobat 4.0)</i> |
| Annots <i>(New in Acrobat 4.0)</i> |
| CreateNotesFile |
| SetBookmarkDest <i>(Removed in Acrobat 4.0)</i> |

Edit Menu Item Names

Methods: [AVMenubarAcquireMenuItemByName](#)

| |
|---|
| Undo |
| endUndoGroup |
| Cut |
| Copy |
| Paste |
| Clear |
| endEditGroup |
| CopyFileToClipboard (<i>Windows only</i>) |
| endOleGroup (<i>Windows only</i>) |
| SelectAll |
| DeselectAll (<i>New in Acrobat 4.0</i>) |
| endSelectGroup (<i>New in Acrobat 4.0</i>) |
| Find (<i>Moved from Tool menu in Acrobat 4.0</i>) |
| FindAgain (<i>Moved from Tool menu in Acrobat 4.0</i>) |
| endFindGroup (<i>Moved from Tool menu in Acrobat 4.0</i>) |
| Properties (<i>Not in Reader; moved from Document menu in 4.0</i>) |
| CropPages (<i>Not in Reader, moved under Document menu in 3.0</i>) |
| RotatePages (<i>Not in Reader, moved under Document menu in 3.0</i>) |
| endPageOpGroup (<i>Not in Reader, moved under Document menu in 3.0</i>) |
| Bookmarks (<i>Not in Reader or Acrobat after 2.1</i>) |
| SetBookmarkDest (<i>Not in Reader; moved to Document menu in Acrobat 3.0</i>) |
| Thumbs (<i>Not in Reader or Acrobat after 2.1</i>) |
| Notes (<i>Not in Reader or Acrobat after 2.1</i>) |
| ImportNotes (<i>Not in Reader; moved to Import submenu in Acrobat 3.0</i>) |
| ExportNotes (<i>Not in Reader; moved to Import submenu in Acrobat 3.0</i>) |
| endObjectsEditGroup (<i>Not in Reader or Acrobat after 2.1</i>) |

Properties *(Not in Reader; moved to Document menu in Acrobat 3.0)*

File Menu Item Names

Methods: [AVMenubarAcquireMenuItemByName](#)

| |
|---|
| Open |
| endFileAccessGroup |
| Close |
| Save <i>(Not in Reader)</i> |
| SaveAs <i>(Not in Reader)</i> |
| Revert <i>(Not in Reader, new in Acrobat 4.0)</i> |
| endSaveGroup <i>(Not in Reader)</i> |
| Import <i>(New in Acrobat 3.0)</i> |
| ImportNotes |
| Export |
| ExportNotes |
| endImportExportGroup <i>(New in Acrobat 3.0)</i> |
| PageSetup |
| Print |
| endPrintGroup |
| DocInfo |
| GeneralInfo |
| endGeneralInfoGroup <i>(New in Acrobat 3.0)</i> |
| OpenInfo <i>(Not in Reader)</i> |
| FontsInfo |
| SecurityInfo |
| PrepressInfo <i>(New in Acrobat 4.0)</i> |
| Prefs |
| GeneralPrefs |
| endGeneralPrefsGroup |
| NotePrefs <i>(Not in Reader)</i> |

| |
|---|
| FullScreenPrefs |
| endDocInfoGroup |
| AdobeOnline <i>(New in Acrobat 4.0)</i> |
| endAdobeOnlineGroup <i>(New in Acrobat 4.0)</i> |
| RecentFile1 <i>(New in Acrobat 3.0)</i> |
| RecentFile2 <i>(New in Acrobat 3.0)</i> |
| RecentFile3 <i>(New in Acrobat 3.0)</i> |
| RecentFile4 <i>(New in Acrobat 3.0)</i> |
| endRecentFileGroup <i>(Windows only)</i> |
| Quit |

Help Menu Item Names

Methods: [AVMenubarAcquireMenuItemByName](#)

| |
|---|
| HelpUserGuide <i>(New in Acrobat 3.0)</i> |
| HelpTutorial <i>(New in Acrobat 3.0)</i> |
| endGuideGroup <i>(Windows only; new in Acrobat 3.0)</i> |
| About <i>(Windows only)</i> |
| AboutExtensions <i>(Windows only)</i> |
| HelpExchange <i>(Removed in Acrobat 4.0)</i> |
| HelpCapture <i>(Removed in Acrobat 4.0)</i> |
| HelpScan <i>(Removed in Acrobat 4.0)</i> |
| endViewerGroup <i>(Removed in Acrobat 4.0)</i> |
| HelpPDFWriter <i>(Removed in Acrobat 4.0)</i> |
| HelpDistiller <i>(Removed in Acrobat 4.0)</i> |
| endCreateGroup <i>(Removed in Acrobat 4.0)</i> |
| HelpSearch <i>(Removed in Acrobat 4.0)</i> |
| HelpCatalog <i>(Removed in Acrobat 4.0)</i> |
| endUsingGroup <i>(Removed in Acrobat 4.0)</i> |

Tool Menu Item Names

Methods: [AVMenubarAcquireMenuItemByName](#)

| |
|--|
| Annots <i>(New in Acrobat 4.0)</i> |
| CreateNotesFile |
| Hand <i>(Removed in Acrobat 4.0)</i> |
| ZoomIn <i>(Removed in Acrobat 4.0)</i> |
| ZoomOut <i>(Removed in Acrobat 4.0)</i> |
| SelectText <i>(Removed in Acrobat 4.0)</i> |
| SelectGraphics <i>(Removed in Acrobat 4.0)</i> |
| Note <i>(Not in Reader) (Removed in Acrobat 4.0)</i> |
| Link <i>(Not in Reader) (Removed in Acrobat 4.0)</i> |
| Thread <i>(Not in Reader) (Removed in Acrobat 4.0)</i> |
| endToolsGroup <i>(Removed in Acrobat 4.0)</i> |
| FindNextNote <i>(Removed in Acrobat 4.0)</i> |

View Menu Item Names

Methods: [AVMenubarAcquireMenuItemByName](#)

| |
|---|
| FullScreen |
| endFullScreenGroup <i>(New in Acrobat 3.0)</i> |
| ZoomViewIn |
| ZoomViewOut |
| ZoomTo |
| endZoomTypeGroup |
| FitPage |
| ActualSize |
| FitWidth |
| FitVisible |
| endFitGroup <i>(New in Acrobat 4.0)</i> |
| SinglePage <i>(New in Acrobat 3.0)</i> |
| OneColumn <i>(New in Acrobat 3.0)</i> |
| TwoColumns <i>(New in Acrobat 3.0)</i> |
| endPageLayoutGroup <i>(Was endPlayLayoutGroup in Acrobat 3.0, corrected in 3.01)</i> |
| UseLocalFonts <i>(New in Acrobat 4.0)</i> |
| ArticleThreads <i>(Removed in Acrobat 4.0)</i> |
| endArticlesGroup <i>(Removed in Acrobat 4.0)</i> |
| PageOnly <i>(Removed in Acrobat 4.0)</i> |
| ShowBookmarks <i>(Removed in Acrobat 4.0)</i> |
| ShowThumbs <i>(Removed in Acrobat 4.0)</i> |

Window Menu Item Names

Methods: [AVMenubarAcquireMenuItemByName](#)

| |
|--|
| Cascade |
| Tile (submenu) |
| TileHorizontal |
| TileVertical |
| CloseAll |
| endWindowLayoutGroup <i>(New in Acrobat 4.0)</i> |
| ShowHideMenuBar <i>(New in Acrobat 3.0)</i> |
| ShowHideToolBar |
| ShowHideTools |
| ShowHideClipboard |
| endShowHideGroup |
| ShowHideBookmarks <i>(New in Acrobat 4.0)</i> |
| ShowHideThumbnails <i>(New in Acrobat 4.0)</i> |
| endShowHideWindowsGroup1 <i>(New in Acrobat 4.0)</i> |
| ShowHideArticles <i>(New in Acrobat 4.0)</i> |
| ShowHideDestinations <i>(New in Acrobat 4.0)</i> |
| WindowMenuSeparator |

Menu Item Names—Tabs

Article Tab Menu Item Names

Methods: [AVMenubarAcquireMenuItemByName](#)

| |
|--|
| MinimizeArticles (<i>New in Acrobat 4.0</i>) |
|--|

Bookmark Tab Menu Item Names

Methods: [AVMenubarAcquireMenuItemByName](#)

| |
|---|
| NewBookmark <i>(Not in Reader; moved from Edit menu in Acrobat 4.0)</i> |
| NewBookmarksFromStructure <i>(New in Acrobat 4.0)</i> |
| endBookmarkOpGroup <i>(Moved from Document menu in Acrobat 4.0)</i> |
| FindCurrentBookmark <i>(New in Acrobat 4.0)</i> |
| endBookmarkOpGroup <i>(Moved from Document menu in Acrobat 4.0)</i> |
| BookmarkShowLocation <i>(New in Acrobat 4.0)</i> |
| MinimizeBookmarks <i>(New in Acrobat 4.0)</i> |

Destination Tab Menu Item Names

Methods: [AVMenubarAcquireMenuItemByName](#)

| |
|---|
| NewDestination (<i>New in Acrobat 4.0</i>) |
| LoadDestination (<i>New in Acrobat 4.0</i>) |

Thumbnail Tab Menu Item Names

Methods: [AVMenubarAcquireMenuItemByName](#)

| |
|---|
| InsertPages <i>(Not in Reader)</i> |
| ExtractPages <i>(Not in Reader)</i> |
| ReplacePages <i>(Not in Reader)</i> |
| DeletePages <i>(Not in Reader)</i> |
| endDocumentOpGroup |
| CropPages <i>(Not in Reader)</i> |
| RotatePages <i>(Not in Reader)</i> |
| endPageOpGroup <i>(Not in Reader)</i> |
| NumberPages <i>(Not in Reader, new in Acrobat 4.0)</i> |
| CreateAllThumbs <i>(Not in Reader; moved from Document menu in Acrobat 4.0)</i> |
| DeleteAllThumbs <i>(Not in Reader; moved from Document menu in Acrobat 4.0)</i> |
| SmallThumbs <i>(New in Acrobat 4.0)</i> |
| LargeThumbs <i>(New in Acrobat 4.0)</i> |

Menu Names

Menus are listed in their order on the menu bar.

Methods: [AVMenubarAcquireMenuByName](#)

| <i>Menu name</i> | <i>Description</i> |
|------------------|---|
| Apple | Apple menu. <i>(Macintosh only)</i> |
| File | File menu. |
| Import | Import submenu of File menu. |
| Export | Export submenu of File menu. |
| DocInfo | Document Info submenu of File menu. |
| Prefs | Preferences submenu of File menu. |
| Edit | Edit menu. |
| Document | Document menu. <i>(New in Acrobat 3.0)</i> |
| Tools | Tools menu. |
| Annots | Annotations submenu of Tools menu. <i>(New in Acrobat 4.0)</i> |
| Extensions | Plug-ins menu. <i>(Optional)</i> |
| View | View menu. |
| Window | Window menu. |
| Tile | Tile submenu of Window menu. |
| Help | Help menu. |
| Bookmarks | Bookmarks tab. <i>(New in Acrobat 4.0)</i> |
| Thumbs | Thumbnails tab. <i>(New in Acrobat 4.0)</i> |
| Articles | Articles tab. <i>(New in Acrobat 4.0)</i> |
| Destinations | Destinations tab. <i>(New in Acrobat 4.0)</i> |
| UsingExtensions | Acrobat Plug-ins Help submenu <i>(Windows only, removed in Acrobat 4.0)</i> |

Replaceable Methods

| |
|--|
| <u>AVAlert</u> |
| <u>AVAppCanQuit</u> |
| <u>AVAppHandleAppleEvent</u> |
| <u>AVDocClose</u> |
| <u>AVDocDoPrint</u> |
| <u>AVDocDoSave</u> |
| <u>AVDocDoSaveAs</u> |
| <u>AVDocDoSaveAsWithParams</u> |
| <u>AVDocOpenFromFileWithParams</u> |
| <u>AVDocPrintPages</u> |
| <u>AVDocPrintPagesWithParams</u> |
| <u>AVPageViewGetNextView</u> |
| <u>PDDocSave</u> |
| <u>PDDocSaveWithParams</u> |
| <u>PDImageSelectAlternate</u> |

Note: These methods are replaceable in Reader—except for [AVDocDoPrint](#), [AVDocDoSave](#), [AVDocDoSaveAs](#), [AVDocDoSaveAsWithParams](#), [PDDocSave](#), and [PDDocSaveWithParams](#).

Selection Types

| | | | |
|-------------------|--|---------------------|---|
| <i>Text</i> | Text in the document | <i>PDTextSelect</i> | PDDocCreateStructTreeRoot |
| <i>Bitmap</i> | Graphics | <i>AVGrafSelect</i> | AVGrafSelectCreate |
| <i>Annotation</i> | Annotation (text annotation, link, and so forth) | | <ol style="list-style-type: none"> 1. Allocate memory using ASmalloc (<code>sizeof(PDAnnot)</code>). The annotation selection server assumes the data was allocated through <i>ASmalloc</i>. 2. Cast the desired annotation by PDAnnotGetCosObj (<i>annot</i>) and copy this Cos object into newly allocated memory. 3. Pass a pointer to this copy. |
| <i>Thumbnail</i> | Thumbnail image | <i>int32</i> | Pass the page number (the first page in a document is page 0). |

Toolbar Button Names

Buttons are listed in their order on the toolbar. In Acrobat 4.0 and later, the toolbar has horizontal and vertical sections.

In Acrobat 4.0 and later, tool buttons may be on toolbars (flyouts) attached to other tool buttons.

Methods: [AVToolBarGetButtonByName](#)

Vertical Toolbar

| <i>Button name</i> | <i>Description</i> |
|---------------------|---|
| Hand | Tool to scroll within the current page. |
| ZoomIn | Tool to increase the zoom factor. Has a flyout. |
| ZoomOut | Tool to decrease the zoom factor. Under ZoomIn flyout. <i>(Added back in Acrobat 4.0)</i> |
| endNavToolsGroup | Separator after navigation tools group. <i>(New in Acrobat 4.0)</i> |
| Select | Tool to select text. Has a flyout. |
| SelectRect | Tool to select columns. Under Select flyout. |
| SelectGraphics | Tool to select graphics. Under Select flyout. |
| Crop | Tool to crop pages. <i>(New in Acrobat 4.0, not in Reader)</i> |
| endSelectToolsGroup | Separator after select tools group. <i>(New in Acrobat 4.0)</i> |
| Note | Tool to create, select, or edit notes. Has a flyout. <i>(Not in Reader)</i> |
| endNoteToolsGroup | Separator after notes tools group. <i>(New in Acrobat 4.0)</i> |
| Link | Tool to create a link or manipulate an existing link. <i>(Not in Reader)</i> |
| Thread | Tool to create an article thread. <i>(New in Acrobat 4.0, not in Reader)</i> |
| endToolsGroup | Separator after tools group. |

Horizontal Toolbar

| <i>Button name</i> | <i>Description</i> |
|---------------------|--|
| AdobeOnline | Visits Adobe on the World Wide Web. <i>(Windows only; new in Acrobat 4.0)</i> |
| endAdobeOnlineGroup | Separator after Adobe Online group. <i>(Windows only; new in Acrobat 4.0)</i> |
| Open | Opens a file. <i>(New in Acrobat 4.0)</i> |
| Save | Saves a file to PDF. <i>(New in Acrobat 4.0)</i> |
| Print | Prints a PDF file. <i>(New in Acrobat 4.0)</i> |
| endFileGroup | Separator after file group. <i>(New in Acrobat 4.0)</i> |
| ToggleSplitter | Shows/Hides the navigation pane. <i>(New in Acrobat 4.0)</i> |
| endSplitterGroup | Separator after Splitter Group. <i>(New in Acrobat 4.0)</i> |
| FirstPage | Goes to the document's first page. |
| PreviousPage | Goes to the previous page in the document. |
| NextPage | Goes to the next page in the document. |
| LastPage | Goes to the document's last page. |
| endPageNavGroup | Separator after page navigation group. |
| GoBack | Goes to the previous view in the view history. |
| GoForward | Goes to the next view in the view history. |
| endPageStackGroup | Separator after history stack group. |
| Zoom100 | Sets the zoom factor to 100% (that is, actual size). |
| FitPage | Sets the zoom factor to fit the entire page into the window. |
| FitVisible | Sets the zoom factor to fit the portion of the page in which drawing appears into the window. |
| endZoomGroup | Separator after zoom group. |
| FindDialog | Displays the Acrobat viewer's Find dialog (not the cross-document search dialog provided by the Search plug-in). |
| endDialogGroup | Separator after dialog group. |

The following tool buttons are available only on an external window toolbar.

| <i>Button name</i> | <i>Description</i> |
|--------------------|---|
| Print | Prints the document. |
| endFileGroup | Separator (not visible in the toolbar). |
| Copy | Copies selection to clipboard. |
| endCopyGroup | Separator (not visible in the toolbar). |
| FindDialog | Finds text. |
| FindAgainDialog | Finds text again. |
| UseNone | Displays only the document, but neither bookmarks nor thumbnail images. <i>(Removed in Acrobat 4.0)</i> |
| UseBookmarks | Displays the document and bookmarks. <i>(Removed in Acrobat 4.0)</i> |
| UseThumbs | Displays the document and thumbnail images. <i>(Removed in Acrobat 4.0)</i> |
| endPageModeGroup | Separator (not visible in the toolbar). <i>(Removed in Acrobat 4.0)</i> |

Tool Names

Methods: [AVAppGetToolByName](#)

| <i>Tool name</i> | <i>Description</i> |
|------------------|--|
| Crop | Tool for cropping pages. <i>(New in Acrobat 4.0)</i> |
| Hand | Hand tool. |
| Note | Tool for making notes. <i>(Not in Reader)</i> |
| Select | Text selection tool. |
| SelectGraphics | Graphics selection tool. <i>(New in Acrobat 2.0)</i> |
| Zoom | Tool for changing the zoom factor. |
| Link | Link creation tool. <i>(Not in Reader)</i> |
| Thread | Thread creation tool. <i>(Not in Reader)</i> |

View Destination Fit Types

| | |
|--------------|--|
| <i>XYZ</i> | Destination specified as upper-left corner point and a zoom factor. |
| <i>Fit</i> | Fits the page into the window, corresponding to the Acrobat viewer's "FitPage" menu item. |
| <i>FitH</i> | Fits the widths of the page into the window, corresponding to the Acrobat viewer's "Fit Width" menu item. |
| <i>FitV</i> | Fits the height of the page into a window. |
| <i>FitR</i> | Fits the rectangle specified by its upper-left and lower-right corner points into the window. |
| <i>FitB</i> | Fits the rectangle containing all visible elements on the page (known as the bounding box) into the window, corresponds to the Acrobat viewer's "Fit Visible" menu item. |
| <i>FitBH</i> | Fits the width of the bounding box into the window. |
| <i>FitBV</i> | Fits the height of the bounding box into the window. |

Notifications

AVAppDidInitialize

```
ACCB1 void ACCB2 AVAppDidInitialize (void *clientData);
```

| | |
|------------------------------|---|
| Description | The Acrobat viewer has finished initializing and is about to enter its event loop. |
| Parameters | <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | AVAppWillQuit |
| Methods | None |

AVAppFrontDocDidChange

```
ACCB1 void ACCB2 AVAppFrontDocDidChange (AVDoc doc,  
void *clientData);
```

Description

The frontmost *AVDoc* has changed.

Note: This notification is not broadcast for external windows, such as OLE applications or PDF files being displayed in Netscape.

Parameters

doc

The document that was brought to the front. *NULL* if there is no frontmost document (for example, the previous frontmost document was just closed).

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[AVDocDidActivate](#)
[AVDocDidDeactivate](#)

Methods

[AVWindowBringToFront](#)
[AVDocOpenFromFile](#)
[AVDocOpenFromFileWithParams](#)
[AVDocOpenFromPDDoc](#)
[AVDocOpenFromPDDocWithParams](#)

AVAppWillCloseAllInternalDocs

```
ACCB1 void ACCB2 AVAppWillCloseAllInternalDocs  
    (void* clientData);
```

| | |
|------------------------------|---|
| Description | All <i>AVDocs</i> will be closed. |
| Parameters | <i>clientData</i> Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using AVAppRegisterNotification . |
| Related Notifications | AVDocDidClose AVDocWillClose |
| Methods | None |

AVAppWillQuit

```
ACCB1 void ACCB2 AVAppWillQuit (void *clientData);
```

Description

The Acrobat viewer is quitting. All documents have been closed. To access or enumerate documents when the application is quitting, replace the [AVAppCanQuit](#) method, access or enumerate documents in your replacement for that procedure, and return *true* to allow the Acrobat viewer to continue quitting.

Parameters

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[AVAppDidInitialize](#)

Methods

None

AVDocDidActivate

```
ACCB1 void ACCB2 AVDocDidActivate (AVDoc doc,  
    void *clientData);
```

Description

An *AVDoc* has activated. At the time this notification is broadcast, it is possible that the window being activated has not yet been brought to the front. For this reason, the [AVAppFrontDocDidChange](#) notification is often more useful.

*Note: [AVAppGetActiveDoc](#) will not necessarily return the *AVDoc* returned in this notification. For instance, if there is an *AVDoc* in an external window (such as a Web browser's) that becomes active, the *AVDoc* returned by this notification won't match what [AVAppGetActiveDoc](#) returns.*

Note: This notification is not broadcast for external windows, such as OLE applications or PDF files being displayed in Netscape.

Parameters

clientData

The document that was activated.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[AVDocDidDeactivate](#)
[AVAppFrontDocDidChange](#)

Methods

[AVDocOpenFromFile](#)
[AVDocOpenFromFileWithParams](#)
[AVDocOpenFromPDDoc](#)
[AVDocOpenFromPDDocWithParams](#)

AVDocDidAddToSelection

```
ACCB1 void ACCB2 AVDocDidAddToSelection (AVDoc doc,  
    ASAtom selType, void *selData, void *addData,  
    void *clientData);
```

| | |
|------------------------------|--|
| Description | The document's selection has been added to or had something removed. |
| Parameters | <p><i>doc</i></p> <p>The document containing the selection.</p> <p><i>selType</i></p> <p>The <i>ASAtom</i> corresponding to the current selection type. See Selection Types for a list of selection types.</p> <p><i>selData</i></p> <p>Pointer to the current selection data <i>after</i> the selection has been added. The format and contents of <i>selData</i> depend on <i>selType</i>.</p> <p><i>addData</i></p> <p>Pointer to the added selection data. The format and contents of <i>addData</i> depend on <i>selType</i>.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | AVDocDidPrint AVDocWillClearSelection |
| Methods | AVDocClearSelection AVDocDeleteSelection AVDocSetSelection |

AVDocDidClickName

```
ACCB1 void ACCB2 AVDocDidClickName (AVDoc doc, CosObj nameObj,  
void* clientData);
```

| | |
|------------------------------|---|
| Description | Acrobat executed an action to go to a named destination. |
| Parameters | <p><i>doc</i></p> <p>The document containing the named destination.</p> <p><i>nameObj</i></p> <p>The Cos object corresponding to the named destination.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | None |
| Methods | None |

AVDocDidClose

```
ACCB1 void ACCB2 AVDocDidClose (AVDoc doc, void *clientData);
```

Description

A document has been closed. Although an *AVDoc* is passed to the routine called by this notification, the document has already been closed but not freed. As a result, all the routine can really do is manipulate any private data in the underlying PDF file at the time this notification occurs.

Parameters

doc

The document that was closed.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[AVDocWillClose](#)

Methods

[AVDocClose](#)

AVDocDidDeactivate

```
ACCB1 void ACCB2 AVDocDidDeactivate (AVDoc doc,  
void *clientData);
```

Description

A document was deactivated.

Note: This notification is not broadcast for external windows, such as OLE applications or PDF files being displayed in Netscape.

Parameters

doc

The document that was deactivated.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[AVDocDidActivate](#)

[AVAppFrontDocDidChange](#)

Methods

[AVDocOpenFromFile](#)

[AVDocOpenFromFileWithParams](#)

[AVDocOpenFromPDDoc](#)

[AVDocOpenFromPDDocWithParams](#)

AVDocDidOpen

```
ACCB1 void ACCB2 AVDocDidOpen (AVDoc doc, ASInt32 error,  
    void *clientData);
```

Description

A document has been opened.

Calling [AVDocClose](#) within this notification is forbidden.

Parameters

doc

The document that was opened.

error

Error code. *error* is set to zero if no errors occurred while opening the file. If an error occurred, *error* contains the error code.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[AVDocWillOpenFromFile](#)

[AVDocWillOpenFromPDDoc](#)

Methods

[AVDocOpenFromFile](#)

[AVDocOpenFromFileWithParams](#)

AVDocDidPerformAction

```
ACCB1 void ACCB2 AVDocDidPerformAction (AVDoc doc,  
    PDAction action, ASInt32 err, void *clientData);
```

Description

An action was performed.

Parameters

doc

The document containing the action that was performed.

action

The action that was performed.

err

Error code. *error* is set to zero if no errors occurred while performing the action. If an error occurred, *error* contains the error code.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[AVDocWillPerformAction](#)

Methods

[AVDocPerformAction](#)

The following methods broadcast this notification if the document has an open action:

[AVDocOpenFromFile](#)

[AVDocOpenFromFileWithParams](#)

[AVDocOpenFromPDDoc](#)

[AVDocOpenFromPDDocWithParams](#)

AVDocDidPrint

```
ACCB1 void ACCB2 AVDocDidPrint (AVDoc doc, void* clientData);
```

Description

This notification is broadcast after printing ends.

Parameters

doc

The document that was printed.

clientData

Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using [AVAppRegisterNotification](#).

Related Notifications

[AVDocWillPrint](#)

Methods

[AVDocPrintPages](#)

[AVDocPrintPagesWithParams](#)

AVDocDidSetSelection

```
ACCB1 void ACCB2 AVDocDidSetSelection (AVDoc doc,  
    ASAtom selType, void *selData, void *clientData);
```

Description

The document's selection has been set.

Parameters

doc

The document whose selection was set.

selType

The *ASAtom* corresponding to the current selection type. See [Selection Types](#) for a list of selection types.

selData

Pointer to the current selection data. The format and contents of *selData* depend on *selType*.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[AVDocDidAddToSelection](#)
[AVDocWillClearSelection](#)

Methods

[AVDocClearSelection](#)
[AVDocDeleteSelection](#)
[AVDocSetSelection](#)

AVDocWantsToDie

```
ACCB1 void ACCB2 AVDocWantsToDie (AVDoc doc, void *clientData);
```

Description

An *AVDoc*'s file stream has been terminated by the [AVDocSetDead](#) method.

Parameters

doc

The *AVDoc* whose file stream has been terminated.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[AVDocDidClose](#)

Methods

[AVDocSetDead](#)

AVDocWillClearSelection

```
ACCB1 void ACCB2 AVDocWillClearSelection (AVDoc doc,  
      ASAtom selType, void *selData, void *clientData);
```

| | |
|------------------------------|--|
| Description | A document's selection is about to be cleared. |
| Parameters | <p><i>doc</i></p> <p>The document whose selection will be cleared.</p> <p><i>selType</i></p> <p>The <i>ASAtom</i> corresponding to the current selection type.</p> <p><i>selData</i></p> <p>Pointer to the current selection data. The format and contents of <i>selData</i> depend on <i>selType</i>.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | AVDocDidAddToSelection AVDocDidPrint |
| Methods | AVDocClearSelection AVDocDeleteSelection AVDocSetSelection |

AVDocWillClose

```
ACCB1 void ACCB2 AVDocWillClose (AVDoc doc, void *clientData);
```

Description

An *AVDoc* will be closed. Neither this notification nor [AVDocDidClose](#) are broadcast if the user selects “Cancel” when prompted to save a modified document as it is being closed.

Parameters

doc

The document that will be closed.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[AVDocDidClose](#)

Methods

[AVDocClose](#)

AVDocWillOpenFromFile

```
ACCB1 void ACCB2 AVDocWillOpenFromFile (ASPathName fileName,  
    ASFileSys fileSys, void *clientData);
```

Description

An *AVDoc* will be opened from a file.

Parameters

fileName

The *ASPathName* for the file that will be opened.

fileSys

The file system responsible for the file to open.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[AVDocDidOpen](#)

Methods

[AVDocOpenFromFile](#)

[AVDocOpenFromFileWithParams](#)

AVDocWillOpenFromPDDoc

```
ACCB1 void ACCB2 AVDocWillOpenFromPDDoc (PDDoc pdDoc,  
void *clientData);
```

Description

An *AVDoc* will be opened from a PDF file.

Parameters

pdDoc

The *PDDoc* for the file that will be opened.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[AVDocDidOpen](#)

Methods

[AVDocOpenFromFile](#)

[AVDocOpenFromFileWithParams](#)

AVDocWillPerformAction

```
ACCB1 void ACCB2 AVDocWillPerformAction (AVDoc doc,  
PDAAction action, void *clientData);
```

| | |
|--------------------|--|
| Description | An action is about to be performed. |
| Parameters | <i>doc</i> The document containing the action that will be performed. <i>action</i> The action that will be performed. <i>clientData</i> Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification . |

Related Notifications [AVDocDidPerformAction](#)

Methods [AVDocPerformAction](#)

The following methods broadcast this notification if the document has an open action:

[AVDocOpenFromFile](#)

[AVDocOpenFromFileWithParams](#)

[AVDocOpenFromPDDoc](#)

[AVDocOpenFromPDDocWithParams](#)

AVDocWillPrint

```
ACCB1 void ACCB2 AVDocWillPrint (AVDoc doc, void* clientData);
```

| | |
|------------------------------|--|
| Description | This notification is broadcast before a document is printed, before any marks are made on the first page. |
| Parameters | <p><i>doc</i></p> <p>The document that is about to be printed.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using AVAppRegisterNotification.</p> |
| Related Notifications | AVDocDidPrint |
| Methods | AVDocPrintPages AVDocPrintPagesWithParams |

AVPageViewDidChange

```
ACCB1 void ACCB2 AVPageViewDidChange (AVPageView pageView,  
    ASInt16 how, void *clientData);
```

Description

The page view has changed. Zero or more of the following events has occurred:

- The page number has changed.
- The zoom factor has changed.
- The window has been resized.
- The page has been scrolled.

Note: If continuous scrolling is turned on (available in Acrobat 3.0 or later) and more than one page is displayed in the AVPageView, alternating mouse clicks in the different pages displayed does not constitute a change to the AVPageView.

Parameters

pageView

The *AVPageView* that has changed.

how

Specifies how the page view did change. *how* is an OR of zero or more of the following (see *AVExpT.h*):

PAGEVIEW_UPDATE_SCROLL — The view has been scrolled.

PAGEVIEW_UPDATE_PAGENUM — The page number has changed.

PAGEVIEW_UPDATE_PAGESIZE — A new view has been created.

PAGEVIEW_UPDATE_ZOOM — The zoom has been changed.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[AVPageViewDidDraw](#)

Methods

[AVPageViewZoomTo](#)
[AVPageViewScrollTo](#)
[AVPageViewScrollToRect](#)
[AVPageViewGoTo](#)
[AVPageViewReadPageDown](#)
[AVPageViewReadPageUp](#)
[AVPageViewEndOperation](#)
[AVPageViewGoBack](#)
[AVPageViewGoForward](#)

AVPageViewDidDraw

```
ACCB1 void ACCB2 AVPageViewDidDraw (AVPageView pageView,  
void *clientData);
```

| | |
|------------------------------|--|
| Description | Redrawing occurred in the page view section of the window. |
| Parameters | <p><i>pageView</i></p> <p>The <i>AVPageView</i> in which drawing occurred.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | AVPageViewDidChange |
| Methods | AVPageViewDrawNow |

PagePDEContentDidChange

```
ACCB1 void ACCB2 PagePDEContentDidChange ( PDPage page ,  
      PDEContent pagesPDEContent ) ;
```

| | |
|------------------------------|---|
| Description | The <i>PDEContent</i> for a page has changed. |
| Parameters | <i>page</i> The page whose <i>PDEContent</i> changed. <i>pagesPDEContent</i> The <i>PDEContent</i> that changed. |
| Related Notifications | PagePDEContentNotCached |
| Methods | PDPagePDEContentWasChanged |

PagePDEContentNotCached

```
ACCB1 void ACCB2 PagePDEContentNotCached ( PDPage page ,  
      PDEContent pagesPDEContent ) ;
```

Description

The *PDEContent* for a page is no longer valid.

This notification is also sent when others change (or delete) a *PDPage*'s contents without using PDFEdit methods. For instance, rotating or deleting a page in the viewer results in this notification being sent.

You should free up any tag data you might have attached to the *PDEContent*.

PDFEdit registers for almost a half dozen different notifications for the different ways Exchange can alter page contents—you may need only this notification.

Parameters

page

The page whose *PDEContent* is no longer valid.

pagesPDEContent

The *PDEContent* that is no longer valid.

Related Notifications

[PagePDEContentDidChange](#)

Methods

[PDDocDeletePages](#)

[PDPageAddCosContents](#)

[PDPageAddCosResource](#)

[PDPageRemoveCosContents](#)

[PDPageRemoveCosResource](#)

[PDPageSetRotate](#)

PDAnnotDidChange

```
ACCB1 void ACCB2 PDAnnotDidChange (PDAnnot annot, ASAtom key,  
    ASInt32 error, void *clientData);
```

Description

An annotation changed in the specified way.

Parameters

annot

The annotation that changed.

key

The *ASAtom* specifying how the annotation changed. The *ASAtom* corresponding to the key that changed in the annotation's Cos dictionary. See Section 6.6 on annotations in the [Portable Document Format Reference Manual](#) for information on the keys.

error

Error code. *error* is set to zero if no errors occurred while changing the annotation. If an error occurred, *error* contains the error code.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDAnnotWillChange](#)

Methods

[PDAnnotNotifyDidChange](#)

[PDAnnotSetRect](#)

[PDTextAnnotSetOpen](#)

[PDAnnotSetColor](#)

[PDAnnotSetTitle](#)

[PDAnnotSetDate](#)

[PDAnnotSetFlags](#)

[PDTextAnnotSetContents](#)

[PDLinkAnnotSetBorder](#)

[PDLinkAnnotSetAction](#)

[AVPageViewSetAnnotLocation](#)

PDAnnotWasCreated

```
ACCB1 void ACCB2 PDAnnotWasCreated (PDAnnot annot, PDPage page,  
void *clientData);
```

| | |
|------------------------------|---|
| Description | An annotation was created. |
| Parameters | <i>annot</i> The annotation that was created. <i>page</i> The page to which the annotation was added. <i>clientData</i> Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification . |
| Related Notifications | PDAnnotDidChange PDAnnotWillChange |
| Methods | PDPageCreateAnnot PDPageAddNewAnnot |

PDAnnotWillChange

```
ACCB1 void ACCB2 PDAnnotWillChange (PDAnnot annot, ASAtom key,  
void *clientData);
```

Description

An annotation will change in the specified way.

Parameters

annot

The annotation that will change.

key

The *ASAtom* specifying how the annotation will change. The *ASAtom* corresponding to the key that changed in the annotation's Cos dictionary. See Section 6.6 on annotations in the [Portable Document Format Reference Manual](#) for information on the keys.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDAnnotDidChange](#)

Methods

[PDAnnotNotifyWillChange](#)

[PDAnnotSetRect](#)

[PDTextAnnotSetOpen](#)

[PDAnnotSetColor](#)

[PDAnnotSetTitle](#)

[PDAnnotSetDate](#)

[PDAnnotSetFlags](#)

[PDTextAnnotSetContents](#)

[PDLinkAnnotSetBorder](#)

[PDLinkAnnotSetAction](#)

[AVPageViewSetAnnotLocation](#)

PDBookmarkDidChange

```
ACCB1 void ACCB2 PDBookmarkDidChange (PDBookmark bookmark,  
ASAtom key, ASInt32 err, void *clientData);
```

| | |
|------------------------------|---|
| Description | A bookmark has been opened/closed, its action has been changed, its title has been changed, or children have been added to it. |
| Parameters | <p><i>bookmark</i></p> <p>The bookmark that will be changed.</p> <p><i>key</i></p> <p>The <i>ASAtom</i> specifying the change that will occur. See PDBookmarkWillChange for a list of the <i>ASAtoms</i> and their meaning.</p> <p><i>err</i></p> <p>Error code. <i>error</i> is set to zero if no errors occurred while changing the bookmark. If an error occurred, <i>error</i> contains the error code.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | PDBookmarkWillChange PDBookmarkDidChangePosition |
| Methods | PDBookmarkSetOpen PDBookmarkSetAction PDBookmarkSetTitle PDBookmarkAddSubtree |

PDBookmarkDidChangePosition

```
ACCB1 void ACCB2 PDBookmarkDidChangePosition  
(PDBookmark bookmark, void *clientData);
```

| | |
|------------------------------|--|
| Description | One or more bookmarks have been moved. |
| Parameters | <i>bookmark</i> The bookmark that was moved. <i>clientData</i> Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification . |
| Related Notifications | PDBookmarkWillChange PDNameTreeNameRemoved |
| Methods | PDBookmarkAddNext PDBookmarkAddPrev PDBookmarkAddNewChild PDBookmarkAddNewSibling PDBookmarkAddChild PDBookmarkAddSubtree |

PDBookmarkDidDestroy

```
ACCB1 void ACCB2 PDBookmarkDidDestroy (PDBookmark bookmark,  
    ASInt32 err, void *clientData);
```

Description

A bookmark was destroyed.

Parameters

bookmark

The bookmark that was destroyed. Because the bookmark has been destroyed, it is not possible to access it.

err

Error code. *error* is set to zero if no errors occurred while destroying the bookmark. If an error occurred, *error* contains the error code.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDBookmarkWillDestroy](#)

Methods

[PDBookmarkDestroy](#)

PDBookmarkDidUnlink

```
ACCB1 void ACCB2 PDBookmarkDidUnlink (PDBookmark bookmark,  
void* clientData);
```

| | |
|------------------------------|---|
| Description | A bookmark was unlinked from the bookmark tree. |
| Parameters | <p><i>bookmark</i></p> <p>The bookmark that was unlinked. Because the bookmark has not been destroyed, it is possible to access it.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | PDBookmarkDidChangePosition |
| Methods | PDBookmarkUnlink |

PDBookmarkWasCreated

```
ACCB1 void ACCB2 PDBookmarkWasCreated (PDBookmark bookmark,  
void *clientData);
```

| | |
|------------------------------|--|
| Description | A bookmark was created. |
| Parameters | <i>bookmark</i> The bookmark that was created. <i>clientData</i> Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification . |
| Related Notifications | PDBookmarkDidDestroy PDBookmarkWillDestroy |
| Methods | PDBookmarkAddNewChild PDBookmarkAddNewSibling |

PDBookmarkWillChange

```
ACCB1 void ACCB2 PDBookmarkWillChange (PDBookmark bookmark,  
ASAtom key, void *clientData);
```

| | |
|------------------------------|--|
| Description | A bookmark will be opened/closed, its action will be changed, its title will be changed, or children will be added to it. |
| Parameters | <p><i>bookmark</i></p> <p>The bookmark that will be changed.</p> <p><i>key</i></p> <p>The <i>ASAtom</i> specifying the change that will occur. <i>key</i> will be an <i>ASAtom</i> corresponding to one of the following:</p> <ul style="list-style-type: none">• <i>Count</i> — Children will be added, or bookmark will be opened/closed.• <i>A</i> — Action will be changed.• <i>Title</i> — Title will be changed. <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | PDBookmarkDidChange PDBookmarkDidChangePosition |
| Methods | PDBookmarkSetOpen PDBookmarkSetAction PDBookmarkSetTitle PDBookmarkAddSubtree |

PDBookmarkWillDestroy

```
ACCB1 void ACCB2 PDBookmarkWillDestroy (PDBookmark bookmark,  
void *clientData);
```

Description

A bookmark will be destroyed.

Parameters

bookmark

The bookmark that will be destroyed.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDBookmarkDidDestroy](#)

Methods

[PDBookmarkDestroy](#)

PDDocDidAddThread

```
ACCB1 void ACCB2 PDDocDidAddThread (PDDoc doc, PDThread thread,  
void *clientData);
```

Description

A thread has been added to a document.

Parameters

doc

The document to which a thread was added.

thread

The thread that was added.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDThreadDidChange](#)

Methods

[PDDocAddThread](#)
[PDDocImportNotes](#)

PDDocDidChangePages

```
ACCB1 void ACCB2 PDDocDidChangePages (PDDoc doc, PDOperation op,  
    ASInt32 fromPage, ASInt32 toPage, ASInt32 error,  
    void *clientData);
```

Description

Pages have been inserted, deleted, moved, or modified.

Parameters

doc

The document in which pages have been changed.

op

The change that was made. *op* will be one of the [PDOperation](#) values.

fromPage

The page number of the first page that was modified.

toPage

The page number of the last page that was modified.

error

Error code. *error* is set to zero if no errors occurred while changing the pages. If an error occurred, *error* contains the error code.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDDocWillChangePages](#)

Methods

[PDDocCreatePage](#)
[PDDocImportNotes](#)
[PDDocReplacePages](#)
[PDDocMovePage](#)
[PDPageAddCosResource](#)
[PDPageRemoveCosResource](#)
[PDPageAddCosContents](#)
[PDPageRemoveCosContents](#)
[PDDocDeletePages](#)
[PDPageSetRotate](#)
[PDPageSetMediaBox](#)
[PDPageSetCropBox](#)

PDDocDidChangeThumbs

```
ACCB1 void ACCB2 PDDocDidChangeThumbs (PDDoc doc,  
void *clientData);
```

Description

Thumbnail images have been added or removed. In addition to the expected ways in which this can occur, it can also occur if pages are inserted into a file.

Parameters

doc

The document in which thumbnail images have been changed.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDDocDidChangePages](#)
[PDDocDidDeletePages](#)
[PDDocDidInsertPages](#)
[PDDocDidReplacePages](#)
[PDDocWillChangePages](#)
[PDDocWillDeletePages](#)
[PDDocWillInsertPages](#)
[PDDocWillReplacePages](#)

Methods

[PDDocCreatePage](#)
[PDDocCreateThumbs](#)
[PDDocDeleteThumbs](#)
[PDDocImportNotes](#)

PDDocDidClose

```
ACCB1 void ACCB2 PDDocDidClose (PDDoc doc, void* clientData);
```

Description

A *PDDoc* closed. A *PDDoc* is closed only if its reference count is zero.

Neither this notification, [PDDocWillClose](#), [AVDocWillClose](#), nor [AVDocDidClose](#) are broadcast if the user selects “Cancel” when prompted to save a modified document as it is being closed.

Parameters

doc

The document that closed.

clientData

Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using [AVAppRegisterNotification](#).

Related Notifications

[PDDocWillClose](#)

Methods

[AVDocClose](#)
[PDDocClose](#)

PDDocDidDeletePages

```
ACCB1 void ACCB2 PDDocDidDeletePages (PDDoc doc,  
    ASInt32 fromPage, ASInt32 toPage, ASInt32 error,  
    void *clientData);
```

| | |
|------------------------------|--|
| Description | One or more pages were deleted. |
| Parameters | <p><i>doc</i></p> <p>The document from which pages were deleted.</p> <p><i>fromPage</i></p> <p>The page number of the first page that was deleted.</p> <p><i>toPage</i></p> <p>The page number of the last page that was deleted.</p> <p><i>error</i></p> <p>Error code. <i>error</i> is set to zero if no errors occurred while deleting the pages. If an error occurred, <i>error</i> contains the error code.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | PDDocWillDeletePages PDDocDidChangePages |
| Methods | PDDocDeletePages |

PDDocDidExportAnnots

```
ACCB1 void ACCB2 PDDocDidExportAnnots (PDDoc doc,  
void* clientData);
```

| | |
|------------------------------|--|
| Description | The annotations of a document were exported. |
| Parameters | <p><i>doc</i></p> <p>The document whose annotations were exported.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using AVAppRegisterNotification.</p> |
| Related Notifications | PDDocDidImportAnnots PDDocWillExportAnnots PDDocWillImportAnnots |
| Methods | PDDocExportNotes |

PDDocDidImportAnnots

```
ACCB1 void ACCB2 PDDocDidImportAnnots (PDDoc doc,  
void* clientData);
```

| | |
|------------------------------|---|
| Description | The annotations from one document were imported into another document. |
| Parameters | <p><i>doc</i></p> <p>The document into which annotations were imported.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using AVAppRegisterNotification.</p> |
| Related Notifications | PDDocDidImportAnnots PDDocWillExportAnnots PDDocWillImportAnnots |
| Methods | PDDocImportCosDocNotes PDDocImportNotes |

PDDocDidInsertPages

```
ACCB1 void ACCB2 PDDocDidInsertPages (PDDoc doc,  
    ASInt32 insertAfterThisPage, PDDoc srcDoc,  
    ASInt32 srcFromPage, ASInt32 srcToPage, ASInt32 error,  
    void *clientData);
```

Description

One or more pages have been inserted.

Parameters

doc

The document into which pages were inserted.

insertAfterThisPage

Page number (in *doc*) after which pages were inserted.

srcDoc

The document that provided the pages that were inserted. This is *NULL* when a new blank page is created and inserted into a document.

srcFromPage

The page number (in *srcDoc*) of the first page that was inserted. Not valid when a new blank page is created and inserted into a document.

srcToPage

The page number (in *srcDoc*) of the last page that was inserted. Not valid when a new blank page is created and inserted into a document.

error

Error code. *error* is set to zero if no errors occurred while inserting the pages. If an error occurred, *error* contains the error code.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDDocWillExportAnnots](#)
[PDDocDidChangePages](#)

Methods

[PDDocCreatePage](#)
[PDDocImportNotes](#)

PDDocDidInsertPagesEx

```
ACCB1 void ACCB2 PDDocDidInsertPagesEx  
(PDDocInsertPagesParams params, void* clientData);
```

| | |
|------------------------------|---|
| Description | Pages were inserted into a document. This notification occurs after the <i>PDDocDidInsertPages</i> notification. |
| Parameters | <i>params</i> A structure describing how pages were inserted. <i>clientData</i> Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using AVAppRegisterNotification . |
| Related Notifications | PDDocDidExportAnnots PDDocWillExportAnnots PDDocWillInsertPagesEx |
| Methods | PDDocCreatePage PDDocInsertPages |

PDDocDidMovePages

```
ACCB1 void ACCB2 PDDocDidMovePages (PDDoc doc,  
    ASInt32 moveAfterThisPage, ASInt32 fromPage, ASInt32 toPage,  
    ASInt32 error, void *clientData);
```

| | |
|------------------------------|--|
| Description | One or more pages were moved. |
| Parameters | <p><i>doc</i></p> <p>The document in which pages were moved.</p> <p><i>moveAfterThisPage</i></p> <p>The page number after which the moved pages were placed.</p> <p><i>fromPage</i></p> <p>The page number of the first page that was moved.</p> <p><i>toPage</i></p> <p>The page number of the last page that was moved.</p> <p><i>error</i></p> <p>Error code. <i>error</i> is set to zero if no errors occurred while moving pages. If an error occurred, <i>error</i> contains the error code.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | PDDocWillMovePages PDDocDidChangePages |
| Methods | PDDocMovePage |

PDDocDidPrintPage

```
ACCB1 void ACCB2 PDDocDidPrintPage (PDDoc doc, ASInt32 page,  
ASStm stm, ASInt32 error, void *clientData);
```

Description

This notification is broadcast once per page that is printed, after all marks have been made on the page. When printing to a PostScript printer, printing commands can also be sent that will be placed on the page after all other marks.

Parameters

doc

The document from which a page was printed.

page

The page number of the page that was printed.

stm

The PostScript print stream when printing to a PostScript printer, and *NULL* when printing to a non-PostScript printer. When printing to a PostScript printer, plug-ins can write printing commands into *stm* (using [ASStmWrite](#)) to add marks to the printed page after all other marks have been made. See [PDDocWillPrintPage](#) for a description of the sequence of operations when printing a page to a PostScript printer.

error

Error code. *error* is set to zero if no errors occurred while printing the page. If an error occurred, *error* contains the error code.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDDocWillPrintPage](#)
[PDDocDidPrintPages](#)

Methods

[AVDocPrintPages](#)

PDDocDidPrintPages

```
ACCB1 void ACCB2 PDDocDidPrintPages (PDDoc doc,  
    ASInt32 fromPage, ASInt32 toPage, ASInt32 error,  
    void *clientData);
```

| | |
|------------------------------|--|
| Description | This notification is broadcast after printing ends. |
| Parameters | <p><i>doc</i></p> <p>The document from which pages were printed.</p> <p><i>fromPage</i></p> <p>The page number of the first page that was printed.</p> <p><i>toPage</i></p> <p>The page number of the last page that was printed.</p> <p><i>error</i></p> <p>Error code. <i>error</i> is set to zero if no errors occurred while printing the pages. If an error occurred, <i>error</i> contains the error code.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | PDDocWillPrintPages PDDocDidPrintPage |
| Methods | AVDocPrintPages |

PDDocDidRemoveThread

```
ACCB1 void ACCB2 PDDocDidRemoveThread (PDDoc doc,  
    ASInt32 index, void *clientData);
```

Description

A thread was removed from a document.

Parameters

doc

The document from which a thread was removed.

index

The index of the thread that was removed. Because the thread has already been removed, it is not possible to access it using *index*.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDDocWillRemoveThread](#)

Methods

[PDDocRemovePageLabel](#)

PDDocDidReplacePages

```
ACCB1 void ACCB2 PDDocDidReplacePages (PDDoc doc,  
    ASInt32 fromPage, ASInt32 toPage, PDDoc srcDoc,  
    ASInt32 srcFromPage, ASInt32 srcToPage, ASInt32 error,  
    void *clientData);
```

Description

One or more pages have been replaced.

Parameters

doc

The document in which pages have been replaced.

fromPage

The page number (in *doc*) of the first page that was replaced.

toPage

The page number (in *doc*) of the last page that was replaced.

srcDoc

The document that provided the replacement pages.

srcFomPage

The page number (in *srcDoc*) of the first replacement page.

srcToPage

The page number (in *srcDoc*) of the last replacement page.

error

Error code. *error* is set to zero if no errors occurred while replacing pages. If an error occurred, *error* contains the error code.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDDocWillReplacePages](#)
[PDDocDidChangePages](#)

Methods

[PDDocReplacePages](#)

PDDocDidSave

```
ACCB1 void ACCB2 PDDocDidSave (PDDoc doc, ASInt32 err,  
    void *clientData);
```

Description

A document has been saved.

Parameters

doc

The document that was saved.

err

Error code. *error* is set to zero if no errors occurred while saving the file. If an error occurred, *error* contains the error code.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDDocWillSave](#)

Methods

[PDDocSave](#)

PDDocPageLabelDidChange

```
ACCB1 void ACCB2 PDDocPageLabelDidChange (PDDoc doc,  
      ASInt32 firstPage, ASInt32 lastPage, void* clientData);
```

| | |
|------------------------------|--|
| Description | A range of pages' labels changed in a <i>PDDoc</i> . |
| Parameters | <i>doc</i> The document containing the pages whose labels changed. <i>firstPage</i> The number of the first page whose label changed. <i>lastPage</i> The number of the last page whose label changed. <i>clientData</i> Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using AVAppRegisterNotification . |
| Related Notifications | None |
| Methods | PDDocSetPageLabel |

PDDocWillChangePages

```
ACCB1 void ACCB2 PDDocWillChangePages (PDDoc doc,  
    PDOperation op, ASInt32 fromPage, ASInt32 toPage,  
    void *clientData);
```

| | |
|------------------------------|---|
| Description | Pages will be inserted, deleted, moved, or modified. |
| Parameters | <p><i>doc</i></p> <p>The document in which pages will be changed.</p> <p><i>op</i></p> <p>The change that will be made. <i>op</i> will be one of the PDOperation values.</p> <p><i>fromPage</i></p> <p>The page number of the first page that will be modified.</p> <p><i>toPage</i></p> <p>The page number of the last page that will be modified.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | PDDocDidChangePages |
| Methods | PDDocDeletePages PDPageSetRotate PDPageSetMediaBox PDPageSetCropBox |

PDDocWillClose

```
ACCB1 void ACCB2 PDDocWillClose (PDDoc doc, void* clientData);
```

Description

A *PDDoc* will be closed. A *PDDoc* is closed only if its reference count is zero.

Neither this notification, [PDDocDidClose](#), [AVDocWillClose](#), nor [AVDocDidClose](#) are broadcast if the user selects “Cancel” when prompted to save a modified document as it is being closed.

Parameters

doc

The document to close.

clientData

Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using [AVAppRegisterNotification](#).

Related Notifications

[PDDocDidClose](#)

Methods

[AVDocClose](#)
[PDDocClose](#)

PDDocWillDeletePages

```
ACCB1 void ACCB2 PDDocWillDeletePages (PDDoc doc,  
    ASInt32 fromPage, ASInt32 toPage, void *clientData);
```

| | |
|------------------------------|--|
| Description | One or more pages will be deleted. |
| Parameters | <p><i>doc</i></p> <p>The document from which pages will be deleted.</p> <p><i>fromPage</i></p> <p>The page number of the first page that will be deleted.</p> <p><i>toPage</i></p> <p>The page number of the last page that will be deleted.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | PDDocDidDeletePages PDDocDidChangePages |
| Methods | PDDocDeletePages |

PDDocWillExportAnnots

```
ACCB1 void ACCB2 PDDocWillExportAnnots (PDDoc doc,  
void* clientData);
```

| | |
|------------------------------|---|
| Description | The annotations of a document will be exported. |
| Parameters | <p><i>doc</i></p> <p>The document whose annotations will be exported.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using AVAppRegisterNotification.</p> |
| Related Notifications | PDDocDidExportAnnots PDDocDidImportAnnots PDDocWillImportAnnots |
| Methods | PDDocExportNotes |

PDDocWillImportAnnots

```
ACCB1 void ACCB2 PDDocWillImportAnnots (PDDoc doc,  
void* clientData);
```

| | |
|------------------------------|--|
| Description | The annotations from one document will be imported into another document. |
| Parameters | <p><i>doc</i></p> <p>The document into which annotations will be imported.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using AVAppRegisterNotification.</p> |
| Related Notifications | PDDocDidExportAnnots PDDocDidImportAnnots PDDocWillImportAnnots |
| Methods | PDDocImportCosDocNotes PDDocImportNotes |

PDDocWillInsertPages

```
ACCB1 void ACCB2 PDDocWillInsertPages (PDDoc doc,  
    ASInt32 insertAfterThisPage, PDDoc srcDoc,  
    ASInt32 srcFromPage, ASInt32 srcToPage, void *clientData);
```

Description

One or more pages will be inserted.

Parameters

doc

The document into which pages will be inserted.

insertAfterThisPage

Page number (in *doc*) after which pages will be inserted.

srcDoc

The document that provides the pages to insert. This is *NULL* when a new blank page is created and inserted into a document.

srcFromPage

The page number (in *srcDoc*) of the first page that will be inserted. Not valid when a new blank page is created and inserted into a document.

srcToPage

The page number (in *srcDoc*) of the last page that will be inserted. Not valid when a new blank page is created and inserted into a document.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDDocDidExportAnnots](#)
[PDDocWillChangePages](#)

Methods

[PDDocCreatePage](#)
[PDDocImportNotes](#)

PDDocWillInsertPagesEx

```
ACCB1 void ACCB2 PDDocWillInsertPagesEx  
(PDDocInsertPagesParams params, void* clientData);
```

| | |
|------------------------------|--|
| Description | Pages will be inserted into a document. This notification occurs after the PDDocWillExportAnnots notification. |
| Parameters | <p><i>params</i></p> <p>A structure describing how pages will be inserted.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using AVAppRegisterNotification.</p> |
| Related Notifications | PDDocDidExportAnnots PDDocDidInsertPagesEx PDDocWillExportAnnots |
| Methods | PDDocCreatePage |

PDDocWillMovePages

```
ACCB1 void ACCB2 PDDocWillMovePages (PDDoc doc,  
    ASInt32 moveAfterThisPage, ASInt32 fromPage, ASInt32 toPage,  
    void *clientData);
```

| | |
|------------------------------|---|
| Description | One or more pages will be moved. |
| Parameters | <p><i>doc</i></p> <p>The document in which pages will be moved.</p> <p><i>moveAfterThisPage</i></p> <p>The page number after which the moved pages will be placed.</p> <p><i>fromPage</i></p> <p>The page number of the first page to move.</p> <p><i>toPage</i></p> <p>The page number of the last page to move.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | PDDocDidMovePages PDDocWillChangePages |
| Methods | PDDocMovePage |

PDDocWillPrintDoc

```
ACCB1 void ACCB2 PDDocWillPrintDoc (PDDoc doc, ASStm stm,  
    ASInt32 psLevel, void *clientData);
```

Description

This notification is broadcast before a document is printed, before any marks are made on the first page. When printing to a PostScript printer, printing commands can also be sent that are placed on the page before any other marks. For example, a **setpagedevice** operator could be placed in the print stream.

Note: Page resources and contents cannot be modified reliably at the time this notification is broadcast.

Parameters

doc

The document that is about to be printed.

stm

The PostScript print stream when printing to a PostScript printer, and *NULL* when printing to a non-PostScript printer. When printing to a PostScript printer, plug-ins can write printing commands into *stm* (using **ASStmWrite**) to add marks to pages before any other marks have been made. In the 2.x Acrobat viewers, the page printing sequence to a PostScript printer is:

page setup (including setpagedevice)...save...

gsave...save...begin... begin...begin...

PDDocWillPrintPage... page contents...

PDDocDidPrintPage... end... end... end...

restore... restore... showpage.

This sequence must not be relied on, and it is to some extent dependent on the printer driver in use. Nevertheless, it is true that by the time the *PDDocWillPrintPage* notification is broadcast, it is too late to perform any **setpagedevice** operations.

psLevel

When printing to a PostScript printer, *psLevel* is either 1 or 2, representing the PostScript level available on the printer. When printing to a non-PostScript printer, *psLevel* is 0. *psLevel* is useful to determine whether or not the output device is a PostScript printer. In addition, when printing to a PostScript printer, *psLevel* is useful to determine the operators that can be sent in any printing code downloaded using the [PDDocWillPrintDoc](#), [PDDocWillPrintPage](#), and [PDDocDidPrintPage](#) notifications.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDDocDidPrintPage](#)
[PDDocWillPrintPages](#)

Methods

[AVDocPrintPages](#)

PDDocWillPrintPage

```
ACCB1 void ACCB2 PDDocWillPrintPage (PDDoc doc, ASInt32 page,  
ASStm stm, void *clientData);
```

Description

This notification is broadcast once per page that is printed, before any marks are made on the page. When printing to a PostScript printer, printing commands can also be sent that will be placed on the page before any other marks.

Note: Page resources and contents cannot be modified reliably at the time this notification is broadcast.

Parameters

doc

The document from which a page is about to be printed.

page

The page number of the page that is about to be printed.

stm

The PostScript print stream when printing to a PostScript printer, and *NULL* when printing to a non-PostScript printer. When printing to a PostScript printer, plug-ins can write printing commands into *stm* (using [ASStmWrite](#)) to add marks to the printed page before any other marks have been made. In the 2.x Acrobat viewers, the page printing sequence to a PostScript printer is:

*page setup (including **setpagedevice**)...save...*

gsave...save...begin... begin...begin...

PDDocWillPrintPage... page contents...

PDDocDidPrintPage... end... end... end...

restore... restore... showpage.

This sequence must not be relied on, and it is to some extent dependent on the printer driver in use. Nevertheless, it is true that by the time the *PDDocWillPrintPage* notification is broadcast, it is too late to perform any **setpagedevice** operations.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDDocDidPrintPage](#)
[PDDocWillPrintPages](#)

Methods

[AVDocPrintPages](#)

PDDocWillPrintPages

```
ACCB1 void ACCB2 PDDocWillPrintPages (PDDoc doc,  
    ASInt32 fromPage, ASInt32 toPage, ASInt32 psLevel,  
    ASBool binaryOK, void *clientData);
```

Description

This notification is broadcast when printing begins, before any pages are printed.

Note: Page resources and contents cannot be modified reliably at the time this notification is broadcast.

Parameters

doc

The document from which pages will be printed.

fromPage

The page number of the first page that will be printed.

toPage

The page number of the last page that will be printed.

psLevel

When printing to a PostScript printer, *psLevel* is either 1 or 2, representing the PostScript level available on the printer. When printing to a non-PostScript printer, *psLevel* is 0. *psLevel* is useful to determine whether or not the output device is a PostScript printer. In addition, when printing to a PostScript printer, *psLevel* is useful to determine the operators that can be sent in any printing code downloaded using the [PDDocWillPrintDoc](#), [PDDocWillPrintPage](#), and [PDDocDidPrintPage](#) notifications.

binaryOK

Valid only when printing to a PostScript printer. Indicates whether or not binary data can be sent.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDDocWillPrintDoc](#)
[PDDocWillPrintPage](#)
[PDDocDidPrintPages](#)

Methods

[AVDocPrintPages](#)

PDDocWillRemoveThread

```
ACCB1 void ACCB2 PDDocWillRemoveThread (PDDoc doc,  
    ASInt32 index, void *clientData);
```

Description

A thread will be removed from a document.

Parameters

doc

The document from which a thread will be removed.

index

The index of the thread that will be removed.
Use [PDDocGetThread](#) to obtain the thread from its index.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDDocDidRemoveThread](#)

Methods

[PDDocRemovePageLabel](#)

PDDocWillReplacePages

```
ACCB1 void ACCB2 PDDocWillReplacePages (PDDoc doc,  
    ASInt32 fromPage, ASInt32 toPage, PDDoc srcDoc,  
    ASInt32 srcFromPage, ASInt32 srcToPage, void *clientData);
```

| | |
|------------------------------|---|
| Description | One or more pages will be replaced. |
| Parameters | <p><i>doc</i></p> <p>The document in which pages will be replaced.</p> <p><i>fromPage</i></p> <p>The page number (in <i>doc</i>) of the first page that will be replaced.</p> <p><i>toPage</i></p> <p>The page number (in <i>doc</i>) of the last page that will be replaced.</p> <p><i>srcDoc</i></p> <p>The document that provides the replacement pages.</p> <p><i>srcFomPage</i></p> <p>The page number (in <i>srcDoc</i>) of the first replacement page.</p> <p><i>srcToPage</i></p> <p>The page number (in <i>srcDoc</i>) of the last replacement page.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | PDDocDidReplacePages PDDocWillChangePages |
| Methods | PDDocReplacePages |

PDDocWillSave

```
ACCB1 void ACCB2 PDDocWillSave (PDDoc doc, void *clientData);
```

Description

A document will be saved.

Parameters

doc

The document that will be saved.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDDocDidSave](#)
[PDDocWillSaveEx](#)

Methods

[PDDocSave](#)
[PDDocSaveWithParams](#)

PDDocWillSaveEx

```
ACCB1 void ACCB2 PDDocWillSaveEx (PDDoc doc,  
    PDDocSaveParams params, void *clientData);
```

| | |
|------------------------------|---|
| Description | A document will be saved. |
| Parameters | <p><i>doc</i></p> <p>The document that will be saved.</p> <p><i>params</i></p> <p>The PDDocSaveParams parameters used when saving a file using PDDocSaveWithParams.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | PDDocDidSave PDDocWillSave |
| Methods | PDDocSave PDDocSaveWithParams |

PDNameTreeNameAdded

```
ACCB1 void ACCB2 PDNameTreeNameAdded (PDNameTree nameTree,  
    CosObj key, CosObj value, void* clientData);
```

Description

An entry was added to a name tree.

Parameters

nameTree

The name tree to which an entry was added.

key

Cos object of the key for the entry. This object is a Cos integer.

value

Cos object for the value associated with *key*.

clientData

Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using [AVAppRegisterNotification](#).

Related Notifications

[PDNameTreeNameRemoved](#)

Methods

[PDNameTreePut](#)

PDNameTreeNameRemoved

```
ACCB1 void ACCB2 PDNameTreeNameRemoved (PDNameTree nameTree,  
    CosObj key, void* clientData);
```

Description

An entry was removed from a name tree.

Parameters

nameTree

The name tree from which an entry was removed.

key

The Cos object of the key for the entry. This object is a Cos integer.

clientData

Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using [AVAppRegisterNotification](#).

Related Notifications

[PDNameTreeNameAdded](#)

Methods

[PDNameTreePut](#)

[PDNameTreeRemove](#)

PDNumTreeNumAdded

```
ACCB1 void ACCB2 PDNumTreeNumAdded (PDNumTree numTree,  
    ASInt32 key, CosObj value, void* clientData);
```

Description

An entry was added to a name tree.

Parameters

nameTree

The name tree to which an entry was added.

key

The key for the entry.

value

Cos object for the value associated with *key*.

clientData

Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using [AVAppRegisterNotification](#).

Related Notifications

[PDNumTreeNumRemoved](#)

Methods

[PDNameTreePut](#)

PDNumTreeNumRemoved

```
ACCB1 void ACCB2 PDNumTreeNumRemoved (PDNumTree numTree,  
    ASInt32 key, void* clientData);
```

| | |
|------------------------------|--|
| Description | An entry was removed from a name tree. |
| Parameters | <i>nameTree</i> The name tree from which an entry was removed. <i>key</i> The key for the entry. <i>clientData</i> Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using AVAppRegisterNotification . |
| Related Notifications | PDNumTreeNumAdded |
| Methods | PDNumTreePut PDNumTreeRemove |

PDPPageContentsDidChange

```
ACCB1 void ACCB2 PDPPageContentsDidChange (PDPPage page,  
void *clientData);
```

| | |
|------------------------------|--|
| Description | The contents of a page have changed and the page will be redrawn. |
| Parameters | <p><i>page</i></p> <p>The page whose contents changed.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | PDPPageContentsDidChangeEx |
| Methods | PDPPageNotifyContentsDidChange PDPPageNotifyContentsDidChangeEx |

PDPPageContentsDidChangeEx

```
ACCB1 void ACCB2 PDPPageContentsDidChangeEx (PDPPage page,  
ASBool invalidateViews, void *clientData);
```

Description

The contents of a page changed. Unlike [PDNameTreeNameRemoved](#), this notification specifies whether or not the page is redrawn immediately.

Parameters

page

The page whose contents changed.

invalidateViews

If *true*, the page is redrawn immediately. If *false*, redrawing is suppressed.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDNameTreeNameRemoved](#)

Methods

[PDPPageNotifyContentsDidChange](#)
[PDPPageNotifyContentsDidChangeEx](#)

PDPageDidAddAnnot

```
ACCB1 void ACCB2 PDPageDidAddAnnot (PDPage page, PDAnnot annot,  
    ASInt32 error, void *clientData);
```

Description

An annotation was added to a page.

Parameters

page

The page to which the annotation was added.

annot

The annotation that was added.

error

Error code. *error* is set to zero if no errors occurred while adding the annotation. If an error occurred, *error* contains the error code.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDPageWillAddAnnot](#)

Methods

[PDPageAddAnnot](#)
[PDPageAddNewAnnot](#)

PDPageDidRemoveAnnot

```
ACCB1 void ACCB2 PDPageDidRemoveAnnot (PDPage page,  
    ASInt32 annotIndex, ASInt32 error, void *clientData);
```

| | |
|------------------------------|--|
| Description | An annotation has been removed from a page. |
| Parameters | <p><i>page</i></p> <p>The page from which an annotation was removed.</p> <p><i>annotIndex</i></p> <p>The index (in the page's annotation array) of the annotation that was removed. Because the annotation has already been removed from the array, it is not possible to access the annotation using <i>annotIndex</i>.</p> <p><i>error</i></p> <p>Error code. <i>error</i> is set to zero if no errors occurred while removing the annotation. If an error occurred, <i>error</i> contains the error code.</p> <p><i>clientData</i></p> <p>Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using AVAppRegisterNotification.</p> |
| Related Notifications | PDPageWillRemoveAnnot |
| Methods | PDPageRemoveAnnot |

PDPageWillAddAnnot

```
ACCB1 void ACCB2 PDPageWillAddAnnot (PDPage page,  
    ASInt32 addAfter, PDAnnot annot, void *clientData);
```

Description

An annotation will be added to a page.

Parameters

page

The page to which the annotation will be added.

addAfter

The index in the page's annotation array after which the annotation will be added.

annot

The annotation that will be added.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDPageDidAddAnnot](#)

Methods

[PDPageAddAnnot](#)

[PDPageAddNewAnnot](#)

PDPageWillRemoveAnnot

```
ACCB1 void ACCB2 PDPageWillRemoveAnnot (PDPage page,  
    ASInt32 annotIndex, void *clientData);
```

Description

An annotation will be removed from a page.

Parameters

page

The page from which an annotation will be removed.

annotIndex

The index (in the page's annotation array) of the annotation that will be removed. Use [PDPageEnumResources](#) to obtain the annotation from its index.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDPageDidRemoveAnnot](#)

Methods

[PDPageRemoveAnnot](#)

PDThreadDidChange

```
ACCB1 void ACCB2 PDThreadDidChange (PDThread thread,  
    void *clientData);
```

Description

A thread was changed.

Parameters

thread

The thread that changed.

clientData

Pointer to a block of user-supplied data that was passed when the plug-in registered for this notification using [AVAppRegisterNotification](#).

Related Notifications

[PDDocDidAddThread](#)

Methods

[PDThreadSetInfo](#)

PSPrintAfterBeginPageSetup

```
ACCB1 void ACCB2 PSPrintAfterBeginPageSetup (PDDoc doc,  
      ASInt32 page, ASStm stm, void* clientData);
```

Description

This notification is broadcast after the beginning of the *Page Setup* (immediately after writing **%%BeginPageSetup**) during the printing of a page to a PostScript printer with the methods [PDFLPrintDoc](#) or [PDDocPrintPages](#). At this point it is possible to use **setpagedevice** and set the graphics state but marks cannot be made on the page.

Parameters

doc

The document from which a page is printed.

page

Page number of the page being printed.

stm

The PostScript print stream. PostScript commands can be added to the print stream using [ASStmWrite](#). Some printer drivers (Windows PS4) may not allow additions to the PostScript stream at this point.

clientData

Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using [AVAppRegisterNotification](#).

Related Notifications

[PSPrintAfterPageTrailer](#)

Methods

[PDDocPrintPages](#)
[PDFLPrintDoc](#)
[PDFLPrintPDF](#)

PSPrintAfterBeginProlog

```
ACCB1 void ACCB2 PSPrintAfterBeginProlog (PDDoc doc, ASStm stm,  
void* clientData);
```

Description

This notification is broadcast after the beginning of the PostScript *Prolog* (immediately after writing **%%BeginPrologue**) during the printing of a document to a PostScript printer with the methods [PDFLPrintDoc](#) or [PDDocPrintPages](#). The Prolog is a set of application-specific procedure definitions that an application may emit in a PostScript stream.

At this point nothing should be added to the PostScript print stream that modifies the graphics state or puts marks on the page. Callers should only emit procset resources.

Parameters

doc

The document that is being printed.

stm

The PostScript print stream. PostScript commands can be added to the print stream using [ASStmWrite](#).

clientData

Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using [AVAppRegisterNotification](#).

Related Notifications

[PSPrintAfterBeginSetup](#)

Methods

[PDDocPrintPages](#)

[PDFLPrintDoc](#)

[PDFLPrintPDF](#)

PSPrintAfterBeginSetup

```
ACCB1 void ACCB2 PSPrintAfterBeginSetup (PDDoc doc, ASStm stm,  
void* clientData);
```

Description

This notification is broadcast after the beginning of the *Document Setup* (immediately after writing **%%BeginSetup**) during the printing of a page to a PostScript printer with the methods [PDFLPrintDoc](#) or [PDDocPrintPages](#). During Document Setup, fonts may be downloaded, **setpagedevice** may be called, procsets may be initialized, the graphics state may be initialized, and so forth.

Parameters

doc

The document that is being printed.

stm

The PostScript print stream. PostScript commands can be added to the print stream using [ASStmWrite](#).

clientData

Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using [AVAppRegisterNotification](#).

Related Notifications

[PSPrintBeforeEndSetup](#)

Methods

[PDDocPrintPages](#)
[PDFLPrintDoc](#)
[PDFLPrintPDF](#)

PSPrintAfterEmitExtGState

```
ACCB1 void ACCB2 PSPrintAfterEmitExtGState (ASStm stm,  
void* clientData);
```

Description

This notification is broadcast after extended graphics state parameters are emitted while printing to a PostScript printer with the methods [PDFLPrintDoc](#) or [PDDocPrintPages](#).

These parameters are typically device-dependent. For information on extended graphics state, see Section 7.14 on extended graphics states in [Portable Document Format Reference Manual](#).

Parameters

stm

The PostScript print stream. PostScript commands can be added to the print stream using [ASStmWrite](#).

clientData

Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using [AVAppRegisterNotification](#).

Related Notifications

None

Methods

[PDDocPrintPages](#)
[PDFLPrintDoc](#)
[PDFLPrintPDF](#)

PSPrintAfterPageTrailer

```
ACCB1 void ACCB2 PSPrintAfterPageTrailer (PDDoc doc, ASStm stm,  
void* clientData);
```

Description

This notification is broadcast after the page trailer is emitted (immediately after writing **%%PageTrailer**) during the printing of a page to a PostScript printer with the methods [PDFLPrintDoc](#) or [PDDocPrintPages](#). At this point it is possible to resolve comments (at end) and emit cleanup code.

Parameters

doc

The document from which a page is printed.

stm

The PostScript print stream. PostScript commands can be added to the print stream using [ASStmWrite](#).

clientData

Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using [AVAppRegisterNotification](#).

Related Notifications

[PSPrintAfterBeginPageSetup](#)

Methods

[PDDocPrintPages](#)

[PDFLPrintDoc](#)

[PDFLPrintPDF](#)

PSPrintAfterTrailer

```
ACCB1 void ACCB2 PSPrintAfterTrailer (PDDoc doc, ASStm stm,  
void* clientData);
```

Description

This notification is broadcast after the DSC trailer is emitted (immediately after writing **%%Trailer**) during the printing of a page to a PostScript printer with the methods [PDFLPrintDoc](#) or [PDDocPrintPages](#). At this point it is possible to resolve comments (at end) and emit cleanup code.

Parameters

doc

The document that is being printed.

stm

The PostScript print stream. PostScript commands can be added to the print stream using [ASStmWrite](#).

clientData

Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using [AVAppRegisterNotification](#).

Related Notifications

[PSPrintBeforeEndSetup](#)

Methods

[PDDocPrintPages](#)

[PDFLPrintDoc](#)

[PDFLPrintPDF](#)

PSPrintBeforeEndComments

```
ACCB1 void ACCB2 PSPrintBeforeEndComments (PDDoc doc,  
      ASStm stm, void* clientData);
```

Description

This notification is broadcast after the DSC page-level comments that apply to all pages have been emitted (immediately before writing **%%EndComments**) during the printing of a page to a PostScript printer with the methods [PDFLPrintDoc](#) or [PDDocPrintPages](#).

Parameters

doc

The document that is being printed.

stm

The PostScript print stream. PostScript commands can be added to the print stream using [ASStmWrite](#).

clientData

Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using [AVAppRegisterNotification](#).

Related Notifications

[PSPrintAfterBeginSetup](#)
[PSPrintBeforeEndSetup](#)

Methods

[PDDocPrintPages](#)
[PDFLPrintDoc](#)
[PDFLPrintPDF](#)

PSPrintBeforeEndSetup

```
ACCB1 void ACCB2 PSPrintBeforeEndSetup (PDDoc doc, ASStm stm,  
void* clientData);
```

Description

This notification is broadcast before the end of *Document Setup* (immediately before writing **%%EndSetup**) during the printing of a page to a PostScript printer with the methods [PDFLPrintDoc](#) or [PDDocPrintPages](#). At this point all of the job level resources and procsets have been added to the print stream.

Parameters

doc

The document that is being printed.

stm

The PostScript print stream. PostScript commands can be added to the print stream using [ASStmWrite](#). Some printer drivers (Windows PS4) may not allow additions to the PostScript stream at this point.

clientData

Pointer to a block of user-supplied data that was passed in by the calling application when this notification was registered for using [AVAppRegisterNotification](#).

Related Notifications

[PSPrintAfterBeginSetup](#)

Methods

[PDDocPrintPages](#)
[PDFLPrintDoc](#)
[PDFLPrintPDF](#)

Errors

Error Systems

ErrSysNone

General error and out of memory

ErrSysCos

CosStore, filters

ErrSysCosSyntax

Cos syntax

ErrSysPDDoc

PDDoc and family, Page tree, bookmarks

ErrSysPDPage

PDPage and family, thumbs, annots

ErrSysPDModel

Global PD

ErrSysAcroView

AcroView

ErrSysPage

Page parsing and RIPping

ErrSysPDFEdit

PDFEdit

ErrSysPDSEdit

PDSEdit

ErrSysFontSvr

Font Server

ErrSysRaster

Rasterizer

ErrSysASFile

ASFile I/O

ErrSysXtnMgr

Extension Manager

ErrSysXtn

Errors registered by plug-ins (see [ASRegisterErrorString](#)) are automatically assigned to this error system.

ErrSysMDSystem

Platform-specific system

ErrSysMDApp

Platform-specific application

Severities

ErrAlways

Always display error, even if others are suppressed.

ErrSilent

Never display a message.

ErrSuppressable

Display a message if the user has not suppressed errors.

ErrWarning

Display a warning.

ErrNoError

No error occurred.

avErrActionExternal

| | |
|--------------------|---|
| Description | This action cannot be performed from within an external window. |
| System | ErrSysAcroView |
| Severity | ErrAlways |
| Platforms | All |

avErrActionFullScreen

| | |
|--------------------|--|
| Description | This action cannot be performed during full-screen mode. |
| System | ErrSysAcroView |
| Severity | ErrAlways |
| Platforms | All |

avErrActionRestricted

| | |
|--------------------|-----------------------------------|
| Description | This action can not be performed. |
| System | ErrSysAcroView |
| Severity | ErrAlways |
| Platforms | All |

avErrBadActionCopy

| | |
|--------------------|---|
| Description | No AVActionCopyProc is registered in an action handler and an action copy request occurred. |
| System | ErrSysAcroView |
| Severity | ErrAlways |
| Platforms | All |

avErrBadAnnotationCopy

| | |
|--------------------|--|
| Description | No AVAnnotHandlerCopyProc registered in an annotation handler and an annotation copy request occurred. |
| System | ErrSysAcroView |
| Severity | ErrAlways |
| Platforms | All |

avErrCantOpenDialog

| | |
|--------------------|---|
| Description | Acrobat can not open this file. There is a modal dialog open. |
| System | ErrSysAcroView |
| Severity | ErrAlways |
| Platforms | All |

avErrCantOpenMoreThanTenDocs

| | |
|--------------------|---|
| Description | No more than ten documents can be opened at a time. |
| System | ErrSysAcroView |
| Severity | ErrAlways |
| Platforms | All |

avErrCantOpenPrinting

| | |
|--------------------|---|
| Description | Acrobat can not open this file while printing another document. |
| System | ErrSysAcroView |
| Severity | ErrAlways |
| Platforms | All |

avErrNoError

| | |
|-------------|--------------------------------|
| Description | No error. |
| System | ErrSysAcroView |
| Severity | ErrAlways |
| Platforms | All |

avErrNoText

| | |
|--------------------|--------------------------------|
| Description | There is no text. |
| System | ErrSysAcroView |
| Severity | ErrAlways |
| Platforms | All |

avErrPrintJobTooBig

| | |
|--------------------|------------------------------------|
| Description | There are too many pages to print. |
| System | ErrSysAcroView |
| Severity | ErrAlways |
| Platforms | All |

avErrTooManyChars

| | |
|--------------------|--|
| Description | There is too much text to display. Cannot display more than 32,000 characters. |
| System | ErrSysAcroView |
| Severity | ErrAlways |
| Platforms | All |

cfMacfBsyErr

| | |
|--------------------|--|
| Description | This file is busy and cannot be deleted. (fBsyErr) |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacdirFulErr

| | |
|--------------------|-------------------------------------|
| Description | The directory is full. (dirFulErr). |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacdskFulErr

| | |
|--------------------|--|
| Description | The document's disk or the disk used for temporary files is full. (dskFulErr). |
| System | ErrSysMDSsystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacdsMemFullErr

| | |
|--------------------|-------------------------------------|
| Description | Out of memory (-26). (dsMemFullErr) |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacdupFNErr

| | |
|--------------------|---|
| Description | Another file already exists under the same name. (dupFNErr) |
| System | ErrSysMDSsystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMaceofErr

| | |
|--------------------|--|
| Description | End of file was reached unexpectedly. (eofErr) |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacfLckdErr

| | |
|--------------------|---------------------------------|
| Description | This file is locked. (fLckdErr) |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacGenPSErr

| | |
|--------------------|--|
| Description | A Postscript error has occurred (-8133). |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacilOAbort

| | |
|--------------------|---|
| Description | An I/O error has occurred (-27). (ilOAbort) |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacioErr

| | |
|--------------------|--|
| Description | A file I/O error has occurred. (ioErr) |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMaciPrSavPFil

| | |
|--------------------|--------------------------------|
| Description | Error saving print file (-1). |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacmemFullErr

| | |
|--------------------|------------------------------------|
| Description | Out of memory (-108). (memFullErr) |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacNoErr

| | |
|--------------------|--------------------------------|
| Description | No error. (noErr). |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacnoMacDskErr

| | |
|--------------------|--|
| Description | This disk is not a Macintosh disk. (noMacDskErr) |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacnsvErr

| | |
|--------------------|--|
| Description | There is no such volume available. (nsvErr). |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacopWrErr

| | |
|--------------------|---|
| Description | This file is already open or in use by another application. (opWrErr) |
| System | ErrSysMDSsystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacpermErr

| | |
|--------------------|--|
| Description | You do not have permission to open this file. (permErr) |
| System | ErrSysMDSsystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacresNotFound

| | |
|--------------------|--|
| Description | Tried to get a nonexistent resource (-192). (resNotFound) |
| System | ErrSysMDSsystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacServerLostConnection

| | |
|--------------------|---|
| Description | This file's server connection has closed down. (-1070) (aspParamErr) |
| System | ErrSysMDSsystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacvLckdErr

| | |
|--------------------|---|
| Description | This volume is locked and cannot be written to. (vLckdErr) |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacvolOffLinErr

| | |
|--------------------|---|
| Description | This file's volume is not available. (volOffLinErr) |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cfMacwrPermErr

| | |
|--------------------|--|
| Description | You do not have permission to write to this file. (wrPermErr) |
| System | ErrSysMDSsystem |
| Severity | ErrAlways |
| Platforms | Macintosh |

cosErrAfterSave

| | |
|--------------------|--|
| Description | Implementation failure: this document is now invalid. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrArrayBounds

| | |
|-------------|--|
| Description | Array out-of-bounds error. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrBadFilterName

| | |
|--------------------|--|
| Description | A stream specifies an unknown filter. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrBadIndex

| | |
|--------------------|--|
| Description | Bad master index. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrBadSyntax

| | |
|--------------------|--|
| Description | Syntax error. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrCancelSave

| | |
|-------------|--|
| Description | A Save operation was canceled. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrCantOpenTempFile

| | |
|--------------------|--|
| Description | A temporary file could not be opened. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrCCFError

| | |
|--------------------|--|
| Description | Error in CCITT fax data filter. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrDCTError

| | |
|--------------------|--|
| Description | Error in JPEG data filter. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrDictKeyNotName

| | |
|--------------------|--|
| Description | Dictionary key must be a name object. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrDocTableFull

| | |
|--------------------|--|
| Description | Cos document table full. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrEncryptionErr

| | |
|--------------------|--|
| Description | <p>Error in encryption filter.</p> <p>This error occurs if the:</p> <ul style="list-style-type: none">• RC4 encryption code fails to initialize• RC4 encryption code fails to update its state when requested• Document's security handler has been changed since the document was last saved, and the length of the <i>cryptData</i> passed internally is too small to hold the new <i>cryptData</i>. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrExpectedArray

| | |
|--------------------|--|
| Description | Expected an array object. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrExpectedBoolean

| | |
|-------------|--|
| Description | Expected an <i>ASBool</i> object. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrExpectedDict

| | |
|--------------------|--|
| Description | Expected a dictionary object. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrExpectedDirect

| | |
|--------------------|--|
| Description | Expected a direct object. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrExpectedName

| | |
|--------------------|--|
| Description | Expected a name object. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrExpectedNull

| | |
|-------------|--|
| Description | Expected a <i>NULL</i> object. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrExpectedNumber

| | |
|--------------------|--|
| Description | Expected a number object. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrExpectedStream

| | |
|--------------------|--|
| Description | Expected a stream object. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrExpectedString

| | |
|--------------------|--|
| Description | Expected a string object. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrInt16OutOfRange

| | |
|--------------------|--|
| Description | A number is out of range. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrInvalidAssignment

| | |
|--------------------|--|
| Description | This direct object already has a container. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrInvalidObj

| | |
|--------------------|--|
| Description | Desired operation cannot be performed on this object. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrListOverflow

| | |
|--------------------|--|
| Description | Operation or data is too complex. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrLZWError

| | |
|--------------------|--|
| Description | Error in LZW data filter. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrNeedFullSave

| | |
|--------------------|--|
| Description | This file must be saved with a full save. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrNeedRebuild

| | |
|--------------------|--|
| Description | The file needs to be repaired. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrNoError

| | |
|-------------|--|
| Description | No error. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrOldLinFormat

| | |
|--------------------|--|
| Description | Obsolete format: treating file as non-linearized. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrReadError

| | |
|--------------------|--|
| Description | Read error. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrRebuildFailed

| | |
|--------------------|--|
| Description | Could not repair file. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrStreamTooShort

| | |
|--------------------|--|
| Description | Stream source is shorter than specified length. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrTempFileFull

| | |
|--------------------|--|
| Description | The temporary file is full or nearly full. Close or save any modified documents. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrTempTooShort

| | |
|--------------------|--|
| Description | Temporary file unexpectedly short. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosErrWriteError

| | |
|--------------------|--|
| Description | Write error. |
| System | ErrSysCos |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

cosSynErrBadArrayDict

| | |
|--------------------|---------------------------------|
| Description | Expected dictionary or array. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrBadCharInHexString

| | |
|--------------------|------------------------------------|
| Description | Non-hex character in a hex string. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrBadDict

| | |
|--------------------|---------------------------------|
| Description | Error reading dictionary. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrBadFRef

| | |
|--------------------|---------------------------------|
| Description | Bad foreign object reference. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrBadLinearized

| | |
|--------------------|-------------------------------------|
| Description | Error reading linearized hint data. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrBadObject

| | |
|--------------------|---------------------------------|
| Description | Error reading object. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrBadObjectLabel

| | |
|--------------------|---|
| Description | <p>Object label badly formatted.</p> <p>This error occurs when the Acrobat viewer tries to read a Cos object in a file. It occurs if the object's:</p> <ul style="list-style-type: none">• Object number is not an integer• Object number does not match the object number the viewer expected• Generation number is not an integer• Generation number does not match the generation number the viewer expected• Object/generation number line does not end with obj. <p>One common cause of this error is incorrect offsets in the xref table. For example, if the offset to object 16 is one byte too high, the viewer sees it as being object number 6—instead of 16—and raises this exception.</p> |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrBadTrailerStart

| | |
|--------------------|--|
| Description | Trailer dictionary start missing '<<'. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrBadXref

| | |
|--------------------|---------------------------------|
| Description | Missing 'xref'. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrBadXrefEntry

| | |
|--------------------|---------------------------------|
| Description | Error reading xref entry. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrBadXrefHeader

| | |
|--------------------|-------------------------------------|
| Description | Xref header should be two integers. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrExtraEndStream

| | |
|--------------------|---------------------------------|
| Description | Unexpected endstream. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrImageNeverEnded

| | |
|--------------------|---------------------------------|
| Description | End of image not found. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrNoEndStream

| | |
|--------------------|---------------------------------|
| Description | Missing endstream. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrNoEOF

| | |
|--------------------|---------------------------------|
| Description | Missing '%%EOF'. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrNoError

| | |
|--------------------|---------------------------------|
| Description | No error. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrNoHeader

| | |
|--------------------|-----------------------------------|
| Description | File does not begin with '%PDF-'. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrNoStartAddress

| | |
|--------------------|--|
| Description | Value of startxref address not an integer. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrNoStartXRef

| | |
|--------------------|-----------------------------------|
| Description | Could not find startxref address. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrPStackUnderflow

| | |
|--------------------|---|
| Description | Parse stack underflow while reading object. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrStringTooLong

| | |
|--------------------|---------------------------------|
| Description | String too long. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrTokenTooLong

| | |
|--------------------|---------------------------------|
| Description | Token too long. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrUnexpectedArray

| | |
|--------------------|---------------------------------|
| Description | Unexpected end of array. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrUnexpectedDict

| | |
|--------------------|---------------------------------|
| Description | Unexpected end of dictionary. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrUnexpectedType

| | |
|--------------------|---------------------------------|
| Description | Unexpected token type. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrUnknownName

| | |
|--------------------|---|
| Description | <p>Unrecognized object name.</p> <p>This error occurs if the Acrobat viewer is reading a Cos object and encounters a token that is not a number, name, or string, and which is not one of the keywords: true, false, null, [,], <<, >>, stream, ID, endobj, startxref, endstream, and R. This error also occurs if the Acrobat viewer encounters one of these keywords in an unexpected place, for example a] without a preceding [.</p> |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrUnknownTokenType

| | |
|--------------------|---------------------------------|
| Description | Unrecognized token type. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

cosSynErrUnterminatedString

| | |
|--------------------|---------------------------------|
| Description | Unterminated string. |
| System | ErrSysCosSyntax |
| Severity | ErrSuppressable |
| Platforms | All |

fileErrAlreadyOpen

| | |
|--------------------|---|
| Description | This file is already open or in use by another application. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrBusy

| | |
|--------------------|--|
| Description | This file is busy and cannot be deleted. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrBytesNotReady

| | |
|--------------------|--|
| Description | Bytes that have been requested by the viewer have not yet arrived. This file error occurs only over slow file systems. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrDirFull

| | |
|--------------------|------------------------------|
| Description | The directory is full. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrDiskFull

| | |
|--------------------|---|
| Description | The document's disk or the disk used for temporary files is full. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrEOF

| | |
|--------------------|---------------------------------------|
| Description | End of file was reached unexpectedly. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrExists

| | |
|--------------------|--|
| Description | Another file already exists under the same name. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrFNF

| | |
|--------------------|------------------------------|
| Description | This file cannot be found. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrGeneral

| | |
|--------------------|------------------------------|
| Description | A file error has occurred. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrIO

| | |
|--------------------|--------------------------------|
| Description | A file I/O error has occurred. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrIOTimeout

| | |
|--------------------|---|
| Description | A file I/O error has occurred. The file connection timed out. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrLocked

| | |
|--------------------|------------------------------|
| Description | This file is locked. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrNoErr

| | |
|--------------------|------------------------------|
| Description | No error. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrNSV

| | |
|--------------------|---|
| Description | The disk containing this file is not available. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrOpenFailed

| | |
|--------------------|------------------------------|
| Description | File open failed. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrPerm

| | |
|--------------------|--------------------------------------|
| Description | You do not have access to this file. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrRead

| | |
|--------------------|---------------------------------|
| Description | A file read error has occurred. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrUserRequestedStop

| | |
|--------------------|------------------------------|
| Description | User requested stop. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrVLocked

| | |
|--------------------|---|
| Description | This disk is locked and cannot be written to. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrWrite

| | |
|--------------------|----------------------------------|
| Description | A file write error has occurred. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fileErrWrPerm

| | |
|--------------------|---|
| Description | You do not have permission to write to this file. |
| System | ErrSysASFile |
| Severity | ErrAlways |
| Platforms | All |

fsErrBadParameter

| | |
|--------------------|--------------------------------------|
| Description | Bad parameter passed to font server. |
| System | ErrSysFontSvr |
| Severity | ErrAlways |
| Platforms | All |

fsErrDownloadAborted

| | |
|--------------------|-------------------------------|
| Description | Font download aborted. |
| System | ErrSysFontSvr |
| Severity | ErrAlways |
| Platforms | All |

fsErrDownloadFailed

| | |
|--------------------|-------------------------------|
| Description | Font download failed. |
| System | ErrSysFontSvr |
| Severity | ErrAlways |
| Platforms | All |

fsErrInitFailed

| | |
|--------------------|--|
| Description | Initialization of the font server module failed. |
| System | ErrSysFontSvr |
| Severity | ErrAlways |
| Platforms | All |

fsErrNeedNewATM

| | |
|--------------------|--|
| Description | A new version of Adobe Type Manager is required. |
| System | ErrSysFontSvr |
| Severity | ErrAlways |
| Platforms | All |

fsErrNoATM

| | |
|--------------------|-----------------------------------|
| Description | Adobe Type Manager was not found. |
| System | ErrSysFontSvr |
| Severity | ErrAlways |
| Platforms | All |

fsErrNoError

| | |
|--------------------|-------------------------------|
| Description | No error. |
| System | ErrSysFontSvr |
| Severity | ErrAlways |
| Platforms | All |

fsErrNoMMFonts

| | |
|--------------------|--------------------------------------|
| Description | No multiple master fonts were found. |
| System | ErrSysFontSvr |
| Severity | ErrAlways |
| Platforms | All |

fsErrNoT1ZapfDingbats

| | |
|--------------------|--|
| Description | The Type 1 font 'Zapf Dingbats' must be installed. |
| System | ErrSysFontSvr |
| Severity | ErrAlways |
| Platforms | All |

genErrBadParm

| | |
|--------------------|----------------------------|
| Description | Bad parameter. |
| System | ErrSysNone |
| Severity | ErrAlways |
| Platforms | All |

genErrBadUnlock

| | |
|--------------------|--|
| Description | Attempt to release an unlocked object. |
| System | ErrSysNone |
| Severity | ErrAlways |
| Platforms | All |

genErrExceptionStackOverflow

| | |
|--------------------|----------------------------|
| Description | Exception stack overflow. |
| System | ErrSysNone |
| Severity | ErrAlways |
| Platforms | All |

genErrGeneral

| | |
|--------------------|-----------------------------|
| Description | An internal error occurred. |
| System | ErrSysNone |
| Severity | ErrAlways |
| Platforms | All |

genErrListOverflow

| | |
|--------------------|-----------------------------------|
| Description | Operation or data is too complex. |
| System | ErrSysNone |
| Severity | ErrAlways |
| Platforms | All |

genErrNoError

| | |
|--------------------|----------------------------|
| Description | No error. |
| System | ErrSysNone |
| Severity | ErrAlways |
| Platforms | All |

genErrNoMemory

| | |
|--------------------|----------------------------|
| Description | Out of memory. |
| System | ErrSysNone |
| Severity | ErrAlways |
| Platforms | All |

genErrResourceLoadFailed

| | |
|--------------------|--|
| Description | Failed to load an application resource (internal error). |
| System | ErrSysNone |
| Severity | ErrAlways |
| Platforms | All |

mdAppCantPrintToPDFWriter

| | |
|--------------------|------------------------------------|
| Description | Cannot print to Acrobat PDFWriter. |
| System | ErrSysMDApp |
| Severity | ErrAlways |
| Platforms | Macintosh |

mdAppErrNoError

| | |
|--------------------|-----------------------------|
| Description | No error. |
| System | ErrSysMDApp |
| Severity | ErrAlways |
| Platforms | Macintosh |

mdAppNoDAsWhilePrint

| | |
|--------------------|--|
| Description | Please close all Desk Accessories before printing. |
| System | ErrSysMDApp |
| Severity | ErrAlways |
| Platforms | Macintosh |

mdAppNoPrinter

| | |
|--------------------|---|
| Description | Printing is not possible until you have chosen a Printer using the Chooser. |
| System | ErrSysMDApp |
| Severity | ErrAlways |
| Platforms | Macintosh |

pageErrArrayLenWrong

| | |
|--------------------|--|
| Description | Array length is out of range. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrBadColorSpace

| | |
|--------------------|--|
| Description | Invalid ColorSpace. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrBadContents

| | |
|-------------|--|
| Description | The page contents object has the wrong type. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrBadDecodeArray

| | |
|--------------------|--|
| Description | Bad decode array. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrBadEGS

| | |
|--------------------|--|
| Description | Invalid Extended Graphics State. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrBadEPSColorSpace

| | |
|--------------------|--|
| Description | An image uses a color space that will not separate correctly in some applications. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrBadForm

| | |
|--------------------|--|
| Description | Invalid Form. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrBadFunction

| | |
|--------------------|--|
| Description | Invalid Function resource. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrBadPattern

| | |
|--------------------|--|
| Description | Invalid Pattern. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrBadType3Font

| | |
|--------------------|--|
| Description | Invalid Type 3 font. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrBadTypeInXTextArray

| | |
|--------------------|--|
| Description | Bad object type within a text operator array. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrColorOutOfRange

| | |
|--------------------|--|
| Description | A color value is out-of-range. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrColorSpaceNotFound

| | |
|--------------------|--|
| Description | Could not find the specified color space. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrEGStateNotFound

| | |
|--------------------|--|
| Description | Could not find the specified Extended Graphics State. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrErrorReadingPage

| | |
|--------------------|--|
| Description | There was an error reading page near the contents. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrFontNotSet

| | |
|--------------------|--|
| Description | Font has not been set. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrFormNotFound

| | |
|--------------------|--|
| Description | Could not find the Form named '%s'. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrErrorParsingImage

| | |
|--------------------|--|
| Description | There was an error while trying to parse an image. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrExpectedEndOfColor

| | |
|--------------------|--|
| Description | Expected end of color space. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrExpectedHexOrASC85

| | |
|--------------------|--|
| Description | Expected ASCIIHexadecimal or ASCII85 string. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrImageExpectedEI

| | |
|--------------------|--|
| Description | Expected 'EI' while parsing an image. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrImageExpectedNumber

| | |
|--------------------|--|
| Description | Expected a number while parsing an image. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrFontNotInResDict

| | |
|--------------------|--|
| Description | Could not find a font in the Resources dictionary—using Helvetica instead. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrFontNotInResources

| | |
|--------------------|--|
| Description | A font is not in the Resources dictionary. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrFormTypeNotAvailable

| | |
|--------------------|--|
| Description | Specified Form type is not supported. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrIllegalOpInPath

| | |
|--------------------|--|
| Description | Illegal operation inside a path. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrIllegalOpInTextObj

| | |
|--------------------|--|
| Description | Illegal operation inside a text object. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrIllegalOpInTextOutline

| | |
|--------------------|--|
| Description | There is an illegal operator inside a text outline object. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrIllegalTextOp

| | |
|--------------------|--|
| Description | Illegal operation outside text object. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrImageTooBig

| | |
|--------------------|--|
| Description | Image in Form, Type 3 font, or Pattern is too big. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrInvalidDash

| | |
|--------------------|--|
| Description | Dash arguments are invalid. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrInvalidGRestore

| | |
|--------------------|--|
| Description | Invalid restore. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrInvalidImageMaskDepth

| | |
|--------------------|--|
| Description | An image is specified as an image mask with more than 1 bit per pixel. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrMissingKey

| | |
|--------------------|--|
| Description | Dictionary is missing the key '%s' |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrMissingResource

| | |
|--------------------|--|
| Description | A resource is missing. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrNoError

| | |
|--------------------|--|
| Description | No error. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrNumberOutOfRange

| | |
|--------------------|--|
| Description | A number value is out of range. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrParseContextError

| | |
|--------------------|--|
| Description | Error while parsing a Form, Type 3 font, or Pattern. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrReadLessImageData

| | |
|--------------------|--|
| Description | Read less image data than expected. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrSeveralParsingErrors

| | |
|-------------|--|
| Description | There were several parsing errors on this page. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrTooFewPathOps

| | |
|--------------------|--|
| Description | Too few operands in path. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrOperandTooLarge

| | |
|--------------------|--|
| Description | An operand is too large. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrOpTooLarge

| | |
|--------------------|--|
| Description | Operand too large. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrPatternNotFound

| | |
|--------------------|--|
| Description | Could not find the Pattern with the specified name. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrPatternTypeNotAvailable

| | |
|--------------------|--|
| Description | Pattern type is not supported. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrReadLessImageColor

| | |
|--------------------|--|
| Description | Read less image color data than expected. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrReadLessImageData

| | |
|--------------------|--|
| Description | Read less image data than expected. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrRecursiveMachine

| | |
|--------------------|--|
| Description | Internal error—machine called recursively. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrTokenTypeNotRec

| | |
|--------------------|--|
| Description | Token type not recognized. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrTooFewArgs

| | |
|-------------|--|
| Description | There were too few arguments. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrTooFewOps

| | |
|-------------|--|
| Description | Too few operands. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrTooManyArgs

| | |
|--------------------|--|
| Description | There were too many arguments. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrUnexpectedOpInDisplay

| | |
|--------------------|--|
| Description | Found an unexpected operator in the display list. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrUnknownColorSpace

| | |
|--------------------|--|
| Description | Unknown color space. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrUnknownFilterName

| | |
|--------------------|--|
| Description | Unknown filter name. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrUnknownXObjectType

| | |
|--------------------|--|
| Description | Unknown <i>XObject</i> type. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrUnrecognizedToken

| | |
|--------------------|--|
| Description | An unrecognized token '%s' was found. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrWrongArgsForSetColor

| | |
|--------------------|--|
| Description | Wrong number of arguments for a setcolor operator. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrWrongNumOpsInCurve

| | |
|--------------------|--|
| Description | A curve operator has the wrong number of operands. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrWrongOperand

| | |
|--------------------|--|
| Description | Wrong operand type—expected type '%s'. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrWrongOpType

| | |
|-------------|--|
| Description | Wrong operand type. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pageErrXObjectNotFound

| | |
|--------------------|--|
| Description | Could not find the <i>XObject</i> named '%s'. |
| System | ErrSysPage |
| Severity | ErrSuppressable ErrSilent |
| Platforms | All |

pdErrAbortNotes

| | |
|--------------------|--|
| Description | Creation of the notes file was canceled. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrAfterSave

| | |
|--------------------|--|
| Description | This document was successfully saved, but an error occurred after saving the document. Please close and reopen the document. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrAlreadyOpen

| | |
|--------------------|--|
| Description | This file is already open. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrATMMemory

| | |
|--------------------|---|
| Description | ATM is running out of memory. Text in font 's' may not render properly. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadAction

| | |
|--------------------|--|
| Description | Invalid action object. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadAnnotation

| | |
|--------------------|--|
| Description | Invalid annotation object. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadAnnotColor

| | |
|--------------------|--|
| Description | Invalid annotation color (only RGB colors are allowed). |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadBaseObj

| | |
|--------------------|--|
| Description | The base pages object is missing or invalid. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadBead

| | |
|--------------------|--|
| Description | Invalid article element. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadBookmark

| | |
|--------------------|--|
| Description | Invalid bookmark object. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadCMap

| | |
|--------------------|--|
| Description | The font contains a bad CMap /Encoding. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadFileSpec

| | |
|--------------------|--|
| Description | Invalid file specification object. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadFont

| | |
|--------------------|--|
| Description | Bad font object or font descriptor object. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadFontBBox

| | |
|--------------------|--|
| Description | The font '%s' contains a bad /BBox. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadFontDescMetrics

| | |
|--------------------|--|
| Description | The font '%s' contains bad /FontDescriptor metrics. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadFontFlags

| | |
|--------------------|--|
| Description | The font '%s' contains bad /Flags. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadFontWidths

| | |
|--------------------|--|
| Description | The font '%s' contains bad /Widths. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadOutlineObj

| | |
|--------------------|--|
| Description | The outlines object is missing or invalid. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadPageObj

| | |
|--------------------|--|
| Description | A page object is missing or invalid. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadPageTree

| | |
|--------------------|--|
| Description | The document's page tree contains an invalid node. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadResMetrics

| | |
|--------------------|--|
| Description | Invalid or corrupt font metrics in the resource file. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadRootObj

| | |
|--------------------|--|
| Description | The root object is missing or invalid. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBadThread

| | |
|--------------------|--|
| Description | Invalid article object. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrBookmarksError

| | |
|--------------------|--|
| Description | There is an error in the bookmarks. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrCancelSave

| | |
|-------------|--|
| Description | A Save operation was canceled. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrCannotMergeWithSubsetFonts

| | |
|--------------------|--|
| Description | Documents contain subset fonts that have the same name and cannot be merged. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrCannotOpenMoreBkMark

| | |
|--------------------|--|
| Description | Cannot open more bookmarks. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrCannotOpenNotes

| | |
|--------------------|--|
| Description | An error occurred while creating the document notes file. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrCannotReopenDoc

| | |
|--------------------|---|
| Description | This document was successfully saved, but an error occurred after saving the document. Close and reopen the document. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrCantUseNewVersion

| | |
|--------------------|--|
| Description | This file contains information not understood by the viewer. It cannot be used for this operation. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrCopyPageFailed

| | |
|--------------------|--|
| Description | The copy of a page failed. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrEmbeddingFont

| | |
|--------------------|--|
| Description | Error while trying to embed a font. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrHostEncodingNotSet

| | |
|--------------------|--|
| Description | The application has not set the host encoding. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrIsFileLocked

| | |
|--------------------|---|
| Description | Unable to open file for writing. It may be locked or unavailable. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrNeedCryptHandler

| | |
|--------------------|--|
| Description | Cannot execute this command on an unsecured document. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrNeedPassword

| | |
|--------------------|--|
| Description | This document requires a password. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrNeedRebuild

| | |
|--------------------|--|
| Description | This file is damaged. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrNoCryptHandler

| | |
|--------------------|---|
| Description | The security plug-in required by this command is unavailable (that is, the necessary security handler is not registered). |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrNoError

| | |
|--------------------|--|
| Description | No error. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrNoNotes

| | |
|--------------------|--|
| Description | This document has no notes. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrNoPDDocForCosDoc

| | |
|--------------------|---|
| Description | There is no <i>PDDoc</i> associated with this <i>CosDoc</i> . |
| System | ErrSysPDDoc |
| Severity | ErrAlways |
| Platforms | All |

pdErrNotEnoughMemoryToOpenDoc

| | |
|--------------------|--|
| Description | There is not enough memory available to open the document. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrNotEnoughSpaceForTempFile

| | |
|--------------------|--|
| Description | There is not enough temporary disk space for this operation. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrOldATMVersion

| | |
|--------------------|--|
| Description | The font '%s' cannot be displayed with the installed version of ATM. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrOldEncryption

| | |
|--------------------|--|
| Description | This viewer cannot decrypt this document. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrOpNotPermitted

| | |
|--------------------|--|
| Description | This operation is not permitted. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrOptMemory

| | |
|--------------------|--|
| Description | There is not enough memory to optimize this file. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrPagesLockedNotDeleted

| | |
|--------------------|--|
| Description | One or more pages are in use and could not be deleted. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrTextStringTooShort

| | |
|--------------------|---|
| Description | There are not enough bytes in text string for multibyte character code. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrThreadProcessing

| | |
|--------------------|--|
| Description | Error while processing articles. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrThumbError

| | |
|--------------------|--|
| Description | Error while processing thumbnail. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrTooManyPagesForInsert

| | |
|--------------------|---|
| Description | Inserting this file would result in a document with too many pages. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrTooManyPagesForOpen

| | |
|--------------------|--|
| Description | This file cannot be opened because it contains too many pages. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrTrySaveAs

| | |
|--------------------|--|
| Description | This file can only be saved using Save As. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrUnableToCloseDueToRefs

| | |
|--------------------|--|
| Description | Unable to close document due to outstanding references. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrUnableToExtractFont

| | |
|--------------------|--|
| Description | Unable to extract the embedded font '%s'. Some characters may not display or print correctly. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrUnableToExtractFontErr

| | |
|--------------------|--|
| Description | Unable to extract embedded font. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrUnableToFindFont

| | |
|--------------------|---|
| Description | Unable to find or create the font '%s'. Some characters may not display or print correctly. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrUnableToOpenDoc

| | |
|--------------------|--|
| Description | This file could not be opened. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrUnableToRead

| | |
|--------------------|--|
| Description | Unable to read file. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrUnableToRecover

| | |
|--------------------|--|
| Description | Unable to recover original file. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrUnableToRenameTemp

| | |
|--------------------|--|
| Description | Unable to rename temporary file to Save As name. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrUnableToWrite

| | |
|--------------------|--|
| Description | Unable to write file. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrUnableToXlateText

| | |
|--------------------|--|
| Description | Some text in the font and character '%s' could not be displayed or printed correctly. The font could not be reencoded. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrUnknownAction

| | |
|--------------------|--|
| Description | Unknown action type. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrUnknownFileType

| | |
|--------------------|---|
| Description | This is not a valid Portable Document File (PDF) document. It cannot be opened. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrUnknownProcsets

| | |
|--------------------|--|
| Description | Information needed to print a page is unavailable. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrWhileRecoverInsertPages

| | |
|--------------------|---|
| Description | There was an error while inserting or extracting pages and another error while trying to recover. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdErrZeroPageFile

| | |
|--------------------|--|
| Description | This file cannot be opened because it has no pages. |
| System | ErrSysPDDoc |
| Severity | ErrSuppressable ErrAlways |
| Platforms | All |

pdPErrBadType3Font

| | |
|--------------------|---|
| Description | Invalid Type 3 font. |
| System | ErrSysPDPage |
| Severity | ErrAlways ErrSuppressable ErrSilent |
| Platforms | All |

pdPErrFormTooComplex

| | |
|--------------------|---|
| Description | Form is too complex to print. |
| System | ErrSysPDPage |
| Severity | ErrAlways ErrSuppressable ErrSilent |
| Platforms | All |

pdPErrNoError

| | |
|--------------------|---|
| Description | No error. |
| System | ErrSysPDPage |
| Severity | ErrAlways ErrSuppressable ErrSilent |
| Platforms | All |

pdPErrPageDimOutOfRange

| | |
|--------------------|---|
| Description | The dimensions of this page are out-of-range. |
| System | ErrSysPDPage |
| Severity | ErrAlways ErrSuppressable ErrSilent |
| Platforms | All |

pdPErrType3TooComplex

| | |
|--------------------|---|
| Description | Type 3 font is too complex to print. |
| System | ErrSysPDPage |
| Severity | ErrAlways ErrSuppressable ErrSilent |
| Platforms | All |

pdPErrUnableToCreateRasterPort

| | |
|--------------------|---|
| Description | Creation of a raster port failed. |
| System | ErrSysPDPage |
| Severity | ErrAlways ErrSuppressable ErrSilent |
| Platforms | All |

pdModErrDuplicateCryptName

| | |
|--------------------|--|
| Description | A security handler is already registered with this name. |
| System | ErrSysPDModel |
| Severity | ErrAlways |
| Platforms | All |

pdModErrEncTablesFailed

| | |
|--------------------|--|
| Description | Unable to initialize font encoding tables. |
| System | ErrSysPDModel |
| Severity | ErrAlways |
| Platforms | All |

pdModErrNoError

| | |
|--------------------|-------------------------------|
| Description | No error. |
| System | ErrSysPDModel |
| Severity | ErrAlways |
| Platforms | All |

pdsErrAlreadyExists

| | |
|--------------------|--|
| Description | There is already a table entry with the same name. |
| System | ErrSysPDSEdit |
| Severity | ErrAlways |
| Platforms | All |

pdsErrBadPDF

| | |
|--------------------|---|
| Description | An incorrect structure was found in the PDF file. |
| System | ErrSysPDSEdit |
| Severity | ErrAlways |
| Platforms | All |

pdsErrCantDo

| | |
|--------------------|---|
| Description | Some software required to perform this operation is not present in this version of Adobe Acrobat. |
| System | ErrSysPDSEdit |
| Severity | ErrAlways |
| Platforms | All |

pdsErrRequiredMissing

| | |
|--------------------|---|
| Description | A required field was missing from a dictionary. |
| System | ErrSysPDSEdit |
| Severity | ErrAlways |
| Platforms | All |

pdsErrWrongTypeEntry

| | |
|--------------------|--------------------------------------|
| Description | Dictionary entry has wrong Cos type. |
| System | ErrSysPDSEdit |
| Severity | ErrAlways |
| Platforms | All |

pdsErrWrongTypeParameter

| | |
|--------------------|---|
| Description | Wrong type parameter supplied to a PDS procedure. |
| System | ErrSysPDSEdit |
| Severity | ErrAlways |
| Platforms | All |

peErrBadBlockHeader

| | |
|--------------------|--|
| Description | Bad block header for Type 1 embedded stream. |
| System | ErrSysPDFEdit |
| Severity | ErrAlways |
| Platforms | All |

peErrCantCreateFontSubset

| | |
|--------------------|--|
| Description | Unable to create embedded font subset. |
| System | ErrSysPDFEdit |
| Severity | ErrAlways |
| Platforms | All |

peErrCantEmbedFont

| | |
|--------------------|---|
| Description | This font is licensed and cannot be embedded. |
| System | ErrSysPDFEdit |
| Severity | ErrAlways |
| Platforms | All |

peErrCantGetAttrs

| | |
|--------------------|------------------------------------|
| Description | Unable to get attributes for font. |
| System | ErrSysPDFEdit |
| Severity | ErrAlways |
| Platforms | All |

peErrCantGetImageDict

| | |
|--------------------|---------------------------------|
| Description | Unable to get image dictionary. |
| System | ErrSysPDFEdit |
| Severity | ErrAlways |
| Platforms | All |

peErrCantGetWidths

| | |
|--------------------|--------------------------------|
| Description | Unable to get widths for font. |
| System | ErrSysPDFEdit |
| Severity | ErrAlways |
| Platforms | All |

peErrCantReadImage

| | |
|--------------------|-------------------------------|
| Description | Unable to read image data. |
| System | ErrSysPDFEdit |
| Severity | ErrAlways |
| Platforms | All |

peErrFontToEmbedNotOnSys

| | |
|--------------------|--|
| Description | Unable to find font to embed on this system. |
| System | ErrSysPDFEdit |
| Severity | ErrAlways |
| Platforms | All |

peErrNoError

| | |
|--------------------|-------------------------------|
| Description | No error. |
| System | ErrSysPDFEdit |
| Severity | ErrAlways |
| Platforms | All |

peErrPStackUnderflow

| | |
|--------------------|---|
| Description | PDFEdit parse stack underflow while reading object. |
| System | ErrSysPDFEdit |
| Severity | ErrAlways |
| Platforms | All |

peErrUnknownPDEColorSpace

| | |
|--------------------|-------------------------------------|
| Description | Unknown <i>PDEColorSpace</i> value. |
| System | ErrSysPDFEdit |
| Severity | ErrAlways |
| Platforms | All |

peErrUnknownResType

| | |
|-------------|---|
| Description | Unknown <i>PDEObject</i> resource type. |
| System | ErrSysPDFEdit |
| Severity | ErrAlways |
| Platforms | All |

peErrWrongPDEObjectType

| | |
|--------------------|----------------------------------|
| Description | Incorrect <i>PDEObject</i> type. |
| System | ErrSysPDFEdit |
| Severity | ErrAlways |
| Platforms | All |

rasErrCreatePort

| | |
|--------------------|-------------------------------------|
| Description | Creation of rasterizer port failed. |
| System | ErrSysRaster |
| Severity | ErrAlways |
| Platforms | All |

rasErrDraw

| | |
|--------------------|------------------------------|
| Description | A rasterizer error occurred. |
| System | ErrSysRaster |
| Severity | ErrAlways |
| Platforms | All |

rasErrInitFailed

| | |
|--------------------|---|
| Description | Initialization of the rasterizer module failed. |
| System | ErrSysRaster |
| Severity | ErrAlways |
| Platforms | All |

rasErrNoError

| | |
|--------------------|------------------------------|
| Description | No error. |
| System | ErrSysRaster |
| Severity | ErrAlways |
| Platforms | All |

WinAccessErr

| | |
|--------------------|--------------------------------|
| Description | Access denied. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

WinBadDiskErr

| | |
|--------------------|--------------------------------|
| Description | Badly formatted disk. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

WinBadDriveErr

| | |
|--------------------|--------------------------------|
| Description | Invalid drive. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

WinBadFileErr

| | |
|--------------------|--------------------------------|
| Description | The file does not exist. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

WinBadHdlErr

| | |
|--------------------|--------------------------------|
| Description | Bad file handle. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

WinBadPathErr

| | |
|--------------------|--------------------------------|
| Description | The path does not exist. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

WinDeviceErr

| | |
|--------------------|--------------------------------|
| Description | Device does not exist. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

WinExistsErr

| | |
|--------------------|--------------------------------|
| Description | File already exists. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

WinGeneralErr

| | |
|--------------------|--------------------------------|
| Description | General failure. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

WinLockErr

| | |
|--------------------|--------------------------------|
| Description | Lock violation. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

WinMemErr

| | |
|--------------------|--------------------------------|
| Description | Not enough memory. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

WinNotDosErr

| | |
|--------------------|--------------------------------|
| Description | Not an MS-DOS disk. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

WinShareErr

| | |
|--------------------|--------------------------------|
| Description | Sharing violation. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

WinTooManyErr

| | |
|--------------------|--------------------------------|
| Description | Too many open files. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

WinWrPermErr

| | |
|--------------------|-----------------------------------|
| Description | You do not have write permission. |
| System | ErrSysMDSystem |
| Severity | ErrAlways |
| Platforms | Windows |

xmErr68KOnly

| | |
|--------------------|--|
| Description | Only the 68K Viewer can load this plug-in. |
| System | ErrSysXtnMgr |
| Severity | ErrAlways |
| Platforms | Macintosh |

xmErrCalledObsoleteProc

| | |
|--------------------|----------------------------------|
| Description | An obsolete function was called. |
| System | ErrSysXtnMgr |
| Severity | ErrAlways |
| Platforms | All |

xmErrCannotReplaceSelector

| | |
|--------------------|---|
| Description | Attempt to replace an unreplaceable selector. |
| System | ErrSysXtnMgr |
| Severity | ErrAlways |
| Platforms | All |

xmErrDuplicatePluginName

| | |
|--------------------|---|
| Description | Two plug-ins are attempting to register with the same name. |
| System | ErrSysXtnMgr |
| Severity | ErrAlways |
| Platforms | All |

xmErrInitializationFailed

| | |
|--------------------|-----------------------------------|
| Description | The plug-in failed to initialize. |
| System | ErrSysXtnMgr |
| Severity | ErrAlways |
| Platforms | All |

xmErrNoError

| | |
|--------------------|------------------------------|
| Description | No error. |
| System | ErrSysXtnMgr |
| Severity | ErrAlways |
| Platforms | All |

xmErrNoPLUGResource

| | |
|--------------------|--|
| Description | The plug-in lacks a PLUG resource of ID 1. |
| System | ErrSysXtnMgr |
| Severity | ErrAlways |
| Platforms | Macintosh |

xmErrNotPrivileged

| | |
|--------------------|--|
| Description | The Acrobat Reader cannot load this plug-in. |
| System | ErrSysXtnMgr |
| Severity | ErrAlways |
| Platforms | All |

xmErrOutOfDateHFT

| | |
|--------------------|---|
| Description | The plug-in was compiled with an out-of-date <i>HFT</i> . |
| System | ErrSysXtnMgr |
| Severity | ErrAlways |
| Platforms | All |

xmErrPluginIncompatible

| | |
|--------------------|--|
| Description | The plug-in is incompatible with this version of the Acrobat viewer. |
| System | ErrSysXtnMgr |
| Severity | ErrAlways |
| Platforms | All |

xmErrPluginLoadFailed

| | |
|--------------------|------------------------------|
| Description | The plug-in failed to load. |
| System | ErrSysXtnMgr |
| Severity | ErrAlways |
| Platforms | All |

xmErrPPCOnly

| | |
|--------------------|--|
| Description | Only the PowerPC Viewer can load this plug-in. |
| System | ErrSysXtnMgr |
| Severity | ErrAlways |
| Platforms | Macintosh |

Macros

A4_GLOBALS

A4_GLOBALS

Description

(Macintosh only) Together with *A5_GLOBALS*, this specifies the address register off of which a plug-in references its globals. They are used only for 68xxx Macintosh plug-ins, not for PowerPC plug-ins. Either *A4_GLOBALS* or *A5_GLOBALS* must be defined to be 1, and the other undefined.

Both *A4_GLOBALS* and *A5_GLOBALS* are set automatically by *PIPrefix.h*.

Header File

PIPrefix.h

Related Macros

[A5_GLOBALS](#)

A5_GLOBALS

A5_GLOBALS

Description

(Macintosh only) Together with *A4_GLOBALS*, this specifies the address register off of which a plug-in references its globals. They are used only for 68xxx Macintosh plug-ins, not for PowerPC plug-ins. Either *A4_GLOBALS* or *A5_GLOBALS* must be defined to be 1, and the other undefined.

Both *A4_GLOBALS* and *A5_GLOBALS* are set automatically by *PIPrefix.h*.

Header File

PIPrefix.h

Related Macros

[A4_GLOBALS](#)

ACCB1

ACCB1

Description

Macro used when declaring callback functions. Its definition is platform-dependent. Use this macro in every callback function you declare.

Use *ACCB1* before the return value in a function declaration, as shown in the Example.

Header File

MacPlatform.h
WINPLTFM.H

Related Macros

[ACCB2](#)
[ACCBPROTO1](#)
[ACCBPROTO2](#)

Example

```
static ACCB1 ASAtom ACCB2
    SnapZoomToolGetType(AVTool tool){
    ...
}
```

ACCBPROTO1

ACCBPROTO1

Description

Macro used when declaring function prototypes. Its definition is platform-dependent. Use this macro in every function prototype you declare.

Use *ACCBPROTO1* before the return value in a function prototype, as shown in the Example.

Header File

MacPlatform.h
WINPLTFM.H

Related Macros

[ACCBPROTO2](#)
[ACCB1](#)
[ACCB2](#)

Example

```
static ACCBPROTO1 void (ACCBPROTO2  
    *DrawImageSelectionCallback)(AVPageView  
    pageView, AVRect* updateRect,  
    void *data);
```


ACCB2

ACCB2

Description

Macro used when declaring callback functions. Its definition is platform-dependent. Use this macro in every callback function you declare.

Use *ACCB2* after the return value in a function declaration, as shown in the Example.

Header File

MacPlatform.h
WINPLTFM.H

Related Macros

[ACCB1](#)
[ACCBPROTO1](#)
[ACCBPROTO2](#)

Example

```
static ACCB1 ASAtom ACCB2
    SnapZoomToolGetType(AVTool tool){
    ...
}
```

ACCBPROTO2

ACCBPROTO2

Description

Macro used when declaring function prototypes. Its definition is platform-dependent. Use this macro in every function prototype you declare.

Use *ACCBPROTO2* after the return value in a function prototype, as shown in the Example.

Header File

MacPlatform.h
WINPLTFM.H

Related Macros

[ACCBPROTO1](#)
[ACCB1](#)
[ACCB2](#)

Example

```
static ACCBPROTO1 void (ACCBPROTO2
    *DrawImageSelectionCallback)(AVPageView
    pageView, AVRect* updateRect,
    void *data);
```

ASCallbackCreateNotification

```
ASCallbackCreateNotification(nsel, proc)
```

| | |
|------------------------|--|
| Description | <p>Creates a callback for the specified notification.</p> <p>Performs compile-time type-checking of the procedure if the DEBUG macro is nonzero.</p> |
| Parameters | <p><i>nsel</i></p> <p>The name of the notification for which the callback is being created. The characters <i>NPROTO</i> are automatically appended to this name. For example, if you specify <i>AVDocDidOpen</i>, it is automatically converted to <i>AVDocDidOpenNPROTO</i>.</p> <p><i>proc</i></p> <p>A pointer to the user-supplied procedure for which a callback is created. The declaration of <i>proc</i> must be consistent with the notification with which it will be used (as specified by <i>nsel</i>).</p> |
| Header File | <i>AVCalls.h</i> |
| Related Macros | ASCallbackCreateReplacement ASCallbackCreateProto ACCB1 ACCB2 DEBUG |
| Related Methods | ASCallbackCreate |
| Example | <pre>myNotification = ASCallbackCreateNotification(AVDocDidOpen, myDocDidOpen);</pre> |

ASCallbackCreateProto

```
ASCallbackCreateProto(funcType, proc)
```

| | |
|------------------------|---|
| Description | <p>Creates a callback for the specified procedure.</p> <p>Performs compile-time type-checking of the procedure if the DEBUG macro is nonzero.</p> |
| Parameters | <p><i>funcType</i></p> <p>The type of function being declared. This is used to allow compile-time type checking against the appropriate function prototype.</p> <p><i>proc</i></p> <p>A pointer to the user-supplied procedure for which a callback is created.</p> |
| Header File | <i>CorCalls.h</i> |
| Related Macros | ASCallbackCreateReplacement ASCallbackCreateNotification ACCB1 ACCB2 DEBUG |
| Related Methods | ASCallbackCreate |
| Example | <pre>snapZoomTool.AdjustCursor = ASCallbackCreateProto(AdjustCursorProcType, SnapZoomAdjustCursor);</pre> |

ASCallbackCreateReplacement

```
ASCallbackCreateReplacement(nsel, proc)
```

Description

Creates a callback appropriate for use in replacing one of the Acrobat viewer's built-in methods. The method being replaced must be one of the [Replaceable Methods](#).

Performs compile-time type-checking of the procedure if the [DEBUG](#) macro is nonzero.

Parameters

nsel

The name of the method for which the replacement callback is being created. The characters *PROTO* are automatically appended to this name. For example, if you specify *AVAppCanQuit*, it is automatically converted to *AVAppCanQuitPROTO*.

proc

A pointer to the user-supplied procedure for which a callback is created. The declaration of *proc* must be consistent with the method being replaced.

Header File

ASCalls.h

Related Macros

[ASCallbackCreateNotification](#)
[ASCallbackCreateProto](#)
[REPLACE](#)
[CALL_REPLACED_PROC](#)
[ACCB1](#)
[ACCB2](#)
[DEBUG](#)

Related Methods

[ASCallbackCreate](#)

Examples

```
ASCallback myAlertCallback;
```

```
myAlertCallback =
    ASCallbackCreateReplacement(AVAlertSEL,
    myAlert);
REPLACE(gAcroViewHFT, AVAlertSEL,
    myAlertCallback);
```

CALL_REPLACED_PROC

CALL_REPLACED_PROC(*hft*, *sel*, *replacer*) (*args...*)

Description

Calls the previous implementation of a replaced method (that is, the code that would have been executed before the method was replaced using [REPLACE](#)).

Parameters

hft

The *HFT* containing the replaced method to execute, for example, *gAcroViewHFT*. See [HFTs](#) for a list of the Acrobat viewer's built-in *HFTs*.

sel

The selector for the replaced method to execute. The name must have the characters *SEL* appended, for example, *AVAlertSEL*.

replacer

The callback whose previous implementation is called. Recall that a method may be replaced more than once, and all replacements for a particular method are kept in a linked list. *proc* must be an element in that linked list, and the entry before *proc* is the one that is called.

args...

The argument list to pass to the procedure being called.

Header File

ASCalls.h

Related Macros

[REPLACE](#)

Related Methods

[HFTGetReplacedEntry](#)

Example

```
CALL_REPLACED_PROC(gAcroViewHFT, AVAlertSEL,  
    myAlertCallback)(iconType, gsm, button1,  
    button2, button3, beep);
```

CastToPDAnnot

CastToPDAnnot (a)

| | |
|-----------------------|--|
| Description | Casts a link annotation or a text annotation to a generic annotation. |
| Parameters | <i>a</i> The link annotation or text annotation to cast. |
| Header File | <i>PDExpT.h</i> |
| Related Macros | <u>CastToPDLinkAnnot</u> <u>CastToPDTextAnnot</u> |

CastToPDLinkAnnot

`CastToPDLinkAnnot(a)`

| | |
|-----------------------|--|
| Description | Casts a generic annotation or a text annotation to a link annotation. |
| Parameters | <i>a</i> The generic annotation or text annotation to cast. |
| Header File | <i>PDExpT.h</i> |
| Related Macros | <u>CastToPDAnnot</u> <u>CastToPDTextAnnot</u> |

CastToPDTextAnnot

`CastToPDTextAnnot (a)`

| | |
|-----------------------|--|
| Description | Casts a link annotation or a generic annotation to a text annotation. |
| Parameters | <i>a</i> The link annotation or generic annotation to cast. |
| Header File | <i>PDExpT.h</i> |
| Related Macros | <u>CastToPDAnnot</u> <u>CastToPDLinkAnnot</u> |

DEBUG

DEBUG

| | |
|-----------------------|--|
| Description | Enables and disables compile-time type-checking in various declarations. Define DEBUG as 1 to enable type-checking (when developing and testing plug-ins) and as 0 to disable type-checking (before shipping your plug-in). |
| Header File | <i>MacPlatform.h</i> <i>WINPLTFM.H</i> |
| Related Macros | <u>ASCallbackCreateNotification</u> <u>ASCallbackCreateProto</u> <u>ASCallbackCreateReplacement</u> |
| Example | <pre>#define DEBUG 1</pre> |

ErrBuildCode

```
ErrBuildCode(xseverity, xsys, xerror)
```

Description

Builds an error code for the specified severity, system, and error number. Error codes are used in exception handling. The [ASRaise](#) method takes an error code for its argument; [ASRegisterErrorString](#) returns an error code.

An error code has three components:

- Severity: none, warning, severe; 4 bits
- System: Cos, PDDoc, and so on; 8 bits
- Error: FileOpen, Syntax, and so on; 16 bits

Parameters

xseverity

Severity of the error. One of [Severities](#).

xsys

Error system. Must be one of [Error Systems](#).

xerror

Error number in the error system, xsys.

Header File

AcroErr.h

Related Macros

[ErrGetCode](#)
[ErrGetSeverity](#)
[ErrGetSignedCode](#)
[ErrGetSystem](#)

Related Methods

[ASRaise](#)
[ASRegisterErrorString](#)

Example

```
myErrorCode = ErrBuildCode(ErrAlways,  
    ErrSysAcroView, avErrActionRestricted);
```

ErrGetCode

`ErrGetCode(xcode)`

Description Gets the error number from an error code.

Parameters *xcode*
Error code.

Header File *AcroErr.h*

Related Macros [ErrBuildCode](#)
[ErrGetSeverity](#)
[ErrGetSignedCode](#)
[ErrGetSystem](#)

Related Methods [ASGetErrorString](#)
[ASGetExceptionErrorCode](#)
[ASRegisterErrorString](#)

Example `errorNumber = ErrGetCode(errorCode);`

ErrGetSeverity

`ErrGetSeverity(xcode)`

| | |
|------------------------|--|
| Description | Gets the error severity from an error code. Returns one of Severities . |
| Parameters | <i>xcode</i> Error code. |
| Header File | <i>AcroErr.h</i> |
| Related Macros | ErrBuildCode ErrGetCode ErrGetSignedCode ErrGetSystem |
| Related Methods | ASGetErrorString ASGetExceptionErrorCode ASRegisterErrorString |
| Example | <pre>errorSeverity = ErrGetSeverity(errorCode);</pre> |

ErrGetSignedCode

`ErrGetSignedCode(xcode)`

Description Gets a signed error number from an error code.

Parameters *xcode*
Error code.

Header File *AcroErr.h*

Related Macros [ErrBuildCode](#)
[ErrGetCode](#)
[ErrGetSeverity](#)
[ErrGetSystem](#)

Related Methods [ASGetErrorString](#)
[ASGetExceptionErrorCode](#)
[ASRegisterErrorString](#)

Example

```
errorSignedNumber =  
    ErrGetSignedCode(errorCode);
```

ErrGetSystem

`ErrGetSystem(xcode)`

| | |
|------------------------|--|
| Description | Gets the error system from an error code. Returns one of Error Systems . |
| Parameters | <i>xcode</i> Error code. |
| Header File | <i>AcroErr.h</i> |
| Related Macros | ErrBuildCode ErrGetCode ErrGetSeverity ErrGetSignedCode |
| Related Methods | ASGetErrorString ASGetExceptionErrorCode ASRegisterErrorString |
| Example | <pre>errorSystem = ErrGetSystem(errorCode);</pre> |

Fixed Numbers

Description A variety of predefined fixed point constants.

Header File *ASExpT.h*

Related Macros [FixedRoundToInt16](#)
[FixedRoundToInt32](#)
[FixedToFloat](#)
[FixedTruncToInt16](#)
[FixedTruncToInt32](#)
[FloatToFixed](#)
[Int16ToFixed](#)
[Int32ToFixed](#)

| | |
|--------------------|-------------------|
| fixedZero | fixedFive |
| fixedHundredth | fixedSix |
| fixedSixteenth | fixedSeven |
| fixedTenth | fixedEight |
| fixedEighth | fixedNine |
| fixedQuarter | fixedTen |
| fixedHalf | fixedEleven |
| fixedThreeQuarters | fixedTwelve |
| fixedPi4 | fixedSixteen |
| fixedSevenEighths | fixedThirtyTwo |
| fixedOne1 | fixedFifty |
| fixedOne | fixedSeventyTwo |
| fixedOneAndQuarter | fixedNinety |
| fixedFourThirds | fixedHundred |
| fixedSqrtTwo | fixedHundredFifty |
| fixedThreeHalves | fixedOneEighty |
| fixedOneAnd3Qtr | fixedTwoSeventy |
| fixedPi2 | fixedFiveHundred |
| fixedGolden | fixedThousand |
| fixedTwo | fixedTenThousand |

| | |
|------------|-----------------------|
| fixedThree | fixedNegativeInfinity |
| fixedFour | fixedPositiveInfinity |

FixedRoundToInt16

`FixedRoundToInt16(f)`

| | |
|-----------------------|--|
| Description | Converts a fixed point number to an <i>ASInt16</i> , rounding the result. |
| Parameters | <i>f</i> The fixed point number to convert. |
| Header File | <i>ASExpT.h</i> |
| Related Macros | Fixed Numbers FixedRoundToInt32 FixedToFloat FixedTruncToInt16 FixedTruncToInt32 FloatToFixed Int16ToFixed Int32ToFixed |

FixedRoundToInt32

`FixedRoundToInt32(f)`

| | |
|-----------------------|--|
| Description | Converts a fixed point number to an <i>ASInt32</i> , rounding the result. |
| Parameters | <i>f</i> The fixed point number to convert. |
| Header File | <i>ASExpT.h</i> |
| Related Macros | Fixed Numbers FixedRoundToInt16 FixedToFloat FixedTruncToInt16 FixedTruncToInt32 FloatToFixed Int16ToFixed Int32ToFixed |

FixedToFloat

`FixedToFloat(f)`

| | |
|-----------------------|---|
| Description | Converts a fixed point number to a floating point number. |
| Parameters | <i>f</i> The fixed point number to convert. |
| Header File | <i>ASExpT.h</i> |
| Related Macros | Fixed Numbers FixedRoundToInt16 FixedRoundToInt32 FixedTruncToInt16 FixedTruncToInt32 FloatToFixed Int16ToFixed Int32ToFixed |

FixedTruncToInt16

`FixedTruncToInt16(f)`

| | |
|-----------------------|--|
| Description | Converts a fixed point number to an <i>ASInt16</i> , truncating the result. |
| Parameters | <i>f</i> The fixed point number to convert. |
| Header File | <i>ASExpT.h</i> |
| Related Macros | Fixed Numbers FixedRoundToInt16 FixedRoundToInt32 FixedToFloat FixedTruncToInt32 FloatToFixed Int16ToFixed Int32ToFixed |

FixedTruncToInt32

FixedTruncToInt32(*f*)

| | |
|-----------------------|--|
| Description | Converts a fixed point number to an <i>ASInt32</i> , truncating the result. |
| Parameters | <i>f</i> The fixed point number to convert. |
| Header File | <i>ASExpT.h</i> |
| Related Macros | Fixed Numbers FixedRoundToInt16 FixedRoundToInt32 FixedToFloat FixedTruncToInt16 FloatToFixed Int16ToFixed Int32ToFixed |

FloatToFixed

FloatToFixed(*f*)

| | |
|-----------------------|---|
| Description | Converts a floating point number to a fixed point number. |
| Parameters | <i>f</i> The floating point number to convert. |
| Header File | <i>ASExpT.h</i> |
| Related Macros | Fixed Numbers FixedRoundToInt16 FixedRoundToInt32 FixedToFloat FixedTruncToInt16 FixedTruncToInt32 Int16ToFixed Int32ToFixed |

Int16ToFixed

```
Int16ToFixed(i)
```

Description Converts an *ASInt16* to a fixed point number.

Parameters *i*
The *ASInt16* to convert.

Header File *ASExpT.h*

Related Macros [Fixed Numbers](#)
[FixedRoundToInt16](#)
[FixedRoundToInt32](#)
[FixedToFloat](#)
[FixedTruncToInt16](#)
[FixedTruncToInt32](#)
[FloatToFixed](#)
[Int32ToFixed](#)

Int32ToFixed

`Int32ToFixed(i)`

Description Converts an *ASInt32* to a fixed point number.

Parameters *i*
The *ASInt32* to convert.

Header File *ASExpT.h*

Related Macros [Fixed Numbers](#)
[FixedRoundToInt16](#)
[FixedRoundToInt32](#)
[FixedToFloat](#)
[FixedTruncToInt16](#)
[FixedTruncToInt32](#)
[FloatToFixed](#)
[Int16ToFixed](#)

MAC_PLATFORM

MAC_PLATFORM

Description

(Macintosh only, previously known as MAC_ENV)
Defined if the plug-in is being compiled for a Macintosh, undefined otherwise. [MAC_PLATFORM](#), [WIN_PLATFORM](#), and [UNIX_PLATFORM](#) should be used by plug-in developers to conditionally compile platform-dependent code.

MAC_PLATFORM is automatically set by the header files.

Header File

Environ.h (based on a value set in *MacPlatform.h*)

Related Macros

[UNIX_PLATFORM](#)
[WIN_PLATFORM](#)

MAC68K

MAC68K

| | |
|-----------------------|--|
| Description | <i>(Macintosh only)</i> Together with POWER_PC , specifies which processor architecture a plug-in is targeted for. Either <i>POWER_PC</i> or <i>MAC68K</i> must be defined as 1, the other as 0. <i>POWER_PC</i> and <i>MAC68K</i> are automatically set by <i>PIPrefix.h</i> . |
| Header File | <i>PIPrefix.h</i> |
| Related Macros | POWER_PC |

PI_ACROSUPPORT_VERSION

PI_ACROSUPPORT_VERSION

| | |
|-----------------------|--|
| Description | Specifies the version of the Acrobat support level <i>HFT</i> . This is automatically set by <i>PIRequir.h</i> . |
| Header File | <i>PIRequir.h</i> |
| Errors | If the <i>HFT</i> version is higher than the viewer loading the plug-in supports, it displays an alert box with the message "There was an error while loading the plug-in '<plug-in name>'. The plug-in is incompatible with this version of the Viewer." |
| Related Macros | <u>PI_ACROVIEW_VERSION</u> <u>PI_CORE_VERSION</u> <u>PI_COS_VERSION</u> <u>PI_MACINTOSH_VERSION</u> <u>PI_PDMODEL_VERSION</u> <u>PI_UNIX_VERSION</u> <u>PI_WIN_VERSION</u> |

PI_ACROVIEW_VERSION

PI_ACROVIEW_VERSION

| | |
|-----------------------|---|
| Description | Specifies the version of the Acrobat viewer level <i>HFT</i> . This is automatically set by <i>PIRequir.h</i> . |
| Header File | <i>PIRequir.h</i> |
| Errors | If the <i>HFT</i> version is higher that the viewer loading the plug-in supports, it displays an alert box with the message "There was an error while loading the plug-in '<plug-in name>'. The plug-in is incompatible with this version of the Viewer." |
| Related Macros | <u>PI_ACROSUPPORT_VERSION</u> <u>PI_CORE_VERSION</u> <u>PI_COS_VERSION</u> <u>PI_MACINTOSH_VERSION</u> <u>PI_PDMODEL_VERSION</u> <u>PI_UNIX_VERSION</u> <u>PI_WIN_VERSION</u> |

PI_CORE_VERSION

PI_CORE_VERSION

| | |
|-----------------------|---|
| Description | Specifies the version of the <i>HFT</i> . This is automatically set by <i>PIRequir.h</i> . |
| Header File | <i>PIRequir.h</i> |
| Errors | If the <i>HFT</i> version is higher than the viewer loading the plug-in supports, it displays an alert box with the message "There was an error while loading the plug-in '<plug-in name>'. The plug-in is incompatible with this version of the Viewer." |
| Related Macros | <u>PI_ACROSUPPORT_VERSION</u> <u>PI_ACROVIEW_VERSION</u> <u>PI_COS_VERSION</u> <u>PI_MACINTOSH_VERSION</u> <u>PI_PDMODEL_VERSION</u> <u>PI_UNIX_VERSION</u> <u>PI_WIN_VERSION</u> |

PI_COS_VERSION

PI_COS_VERSION

| | |
|-----------------------|--|
| Description | Specifies the version of the Cos level <i>HFT</i> . This is automatically set by <i>PIRequir.h</i> . |
| Header File | <i>PIRequir.h</i> |
| Errors | If the <i>HFT</i> version is higher than the viewer loading the plug-in supports, it displays an alert box with the message "There was an error while loading the plug-in '<plug-in name>'. The plug-in is incompatible with this version of the Viewer." |
| Related Macros | <u>PI_ACROSUPPORT_VERSION</u> <u>PI_ACROVIEW_VERSION</u> <u>PI_CORE_VERSION</u> <u>PI_MACINTOSH_VERSION</u> <u>PI_PDMODEL_VERSION</u> <u>PI_UNIX_VERSION</u> <u>PI_WIN_VERSION</u> |

PI_MACINTOSH_VERSION

PI_MACINTOSH_VERSION

| | |
|-----------------------|--|
| Description | Specifies the version of the Macintosh-only methods <i>HFT</i> . This is automatically set by <i>PIRequir.h</i> . |
| Header File | <i>PIRequir.h</i> |
| Errors | If the <i>HFT</i> version is higher than the viewer loading the plug-in supports, it displays an alert box with the message "There was an error while loading the plug-in '<plug-in name>'. The plug-in is incompatible with this version of the Viewer." |
| Related Macros | <u>PI_ACROSUPPORT_VERSION</u> <u>PI_ACROVIEW_VERSION</u> <u>PI_CORE_VERSION</u> <u>PI_COS_VERSION</u> <u>PI_PDMODEL_VERSION</u> <u>PI_UNIX_VERSION</u> <u>PI_WIN_VERSION</u> |

PI_PDMODEL_VERSION

PI_PDMODEL_VERSION

| | |
|-----------------------|--|
| Description | Specifies the version of the PD level <i>HFT</i> . This is automatically set by <i>PIRequir.h</i> . |
| Header File | <i>PIRequir.h</i> |
| Errors | If the <i>HFT</i> version is higher than the viewer loading the plug-in supports, it displays an alert box with the message "There was an error while loading the plug-in '<plug-in name>'. The plug-in is incompatible with this version of the Viewer." |
| Related Macros | <u>PI_ACROSUPPORT_VERSION</u> <u>PI_ACROVIEW_VERSION</u> <u>PI_CORE_VERSION</u> <u>PI_COS_VERSION</u> <u>PI_MACINTOSH_VERSION</u> <u>PI_UNIX_VERSION</u> <u>PI_WIN_VERSION</u> |

PI_UNIX_VERSION

PI_UNIX_VERSION

| | |
|-----------------------|---|
| Description | Specifies the version of the UNIX-only methods <i>HFT</i> . This is automatically set by <i>PIRequir.h</i> . |
| Header File | <i>PIRequir.h</i> |
| Errors | If the <i>HFT</i> version is higher than the viewer loading the plug-in supports, it displays an alert box with the message "There was an error while loading the plug-in '<plug-in name>'. The plug-in is incompatible with this version of the Viewer." |
| Related Macros | <u>PI_ACROSUPPORT_VERSION</u> <u>PI_ACROVIEW_VERSION</u> <u>PI_CORE_VERSION</u> <u>PI_COS_VERSION</u> <u>PI_MACINTOSH_VERSION</u> <u>PI_PDMODEL_VERSION</u> <u>PI_WIN_VERSION</u> |

PI_WIN_VERSION

PI_WIN_VERSION

| | |
|-----------------------|--|
| Description | Specifies the version of the Windows-only methods <i>HFT</i> . This is automatically set by <i>PIRequir.h</i> . |
| Header File | <i>PIRequir.h</i> |
| Errors | If the <i>HFT</i> version is higher than the viewer loading the plug-in supports, it displays an alert box with the message "There was an error while loading the plug-in '<plug-in name>'. The plug-in is incompatible with this version of the Viewer." |
| Related Macros | <u>PI_ACROSUPPORT_VERSION</u> <u>PI_ACROVIEW_VERSION</u> <u>PI_CORE_VERSION</u> <u>PI_COS_VERSION</u> <u>PI_MACINTOSH_VERSION</u> <u>PI_PDMODEL_VERSION</u> <u>PI_UNIX_VERSION</u> |

PLATFORM

PLATFORM

Description

Defines the platform-specific header file. Must be “*MacPlatform.h*” in Mac OS, “*WINPLTFM.H*” in Windows. *PLATFORM* is automatically set by the header file.

Header File

PIPrefix.h (Macintosh)
ENVIRON.H (Windows)

POWER_PC

POWER_PC

| | |
|-----------------------|--|
| Description | <i>(Macintosh only)</i> Together with MAC68K , specifies which processor architecture a plug-in is targeted for. Either <i>POWER_PC</i> or <i>MAC68K</i> must be defined as 1, the other as 0. <i>POWER_PC</i> and <i>MAC68K</i> are automatically set by <i>PIPrefix.h</i> . |
| Header File | <i>PIPrefix.h</i> |
| Related Macros | MAC68K |

PRODUCT

PRODUCT

Description

Defines the platform-specific header file. Must be "*Plugin.h*" in Mac OS and Windows.

PRODUCT is automatically set by the header file.

Header File

PIPrefix.h (Macintosh)

ENVIRON.H (Windows)

REPLACE

```
REPLACE(hft, sel, proc)
```

Description

Replaces one of the Acrobat viewer's built-in methods. The method being replaced must be one of the [Replaceable Methods](#). The method's *HFTEntryReplaceable* flag is automatically set, allowing it to be subsequently replaced.

All plug-ins, and the Acrobat viewer itself, share a single copy of each *HFT*. As a result, when a plug-in replaces the implementation of a method, all other plug-ins and the Acrobat viewer also use the new implementation of that method. In addition, once a method's implementation has been replaced, there is no way to remove the new implementation without restarting the Acrobat viewer.

Parameters

hft

The *HFT* containing the method to replace, for example, *gAcroViewHFT*. See [HFTs](#) for a list of the Acrobat viewer's built-in *HFTs*.

sel

The selector for the method to replace. The name must have the characters *SEL* appended, for example, *AVAlertSEL*.

proc

A callback containing the replacement method. The callback must have been created using [ASCallbackCreateReplacement](#).

Header File

ASCalls.h

Related Macros

[ASCallbackCreateReplacement](#)
[CALL_REPLACED_PROC](#)

Related Methods

[HFTReplaceEntry](#)

Examples

```
myAlertCallback =
    ASCallbackCreateReplacement(AVAlertSEL,
    myAlert);
REPLACE(gAcroViewHFT, AVAlertSEL,
myAlertCallback);
```

UNIX_PLATFORM

UNIX_PLATFORM

| | |
|-----------------------|---|
| Description | <p>(<i>UNIX only</i>) Defined if the plug-in is being compiled for a UNIX platform, undefined otherwise.</p> <p>MAC_PLATFORM, WIN_PLATFORM, and UNIX_PLATFORM should be used by plug-in developers to conditionally compile platform-dependent code.</p> <p><i>UNIX_PLATFORM</i> must be defined in the arguments to the C compiler. The make files for the sample plug-ins in the Acrobat SDK do this automatically.</p> |
| Header File | <p><i>Environ.h</i> (based on a value set in <i>UNIXPlatform.h</i>)</p> |
| Related Macros | <p>MAC_PLATFORM WIN_PLATFORM</p> |

WIN_PLATFORM

WIN_PLATFORM

Description

(Windows only, previously known as WIN_ENV)
Defined if the plug-in is being compiled for a Windows machine, undefined otherwise. [MAC_PLATFORM](#), [WIN_PLATFORM](#), and [UNIX_PLATFORM](#) should be used by plug-in developers to conditionally compile platform-dependent code.

WIN_PLATFORM must be defined in the arguments to the C compiler. The make files for the sample plug-ins in the Acrobat SDK do this automatically.

Header File

Environ.h (based on a value set in *WinPltfrm.h*)

Related Macros

[MAC_PLATFORM](#)
[UNIX_PLATFORM](#)

Objects

AS Layer

Acrobat Support layer. Platform-independent objects and utility functions used throughout the API.

ASAtom

A hashed token used in place of strings to optimize performance (it is much faster to compare *ASAtoms* than strings).

Obtaining:

[ASAtomFromString](#)
[AVActionHandlerGetType](#)
[AVAppGetName](#)
[AVDocGetSelectionType](#)
[AVMenuGetName](#)
[AVMenuItemGetName](#)
[AVToolButtonGetIcon](#)
[AVToolGetType](#)
[CosNameValue](#)
[PDActionGetSubtype](#)
[PDAnnotGetSubtype](#)
[PDFFileSpecGetFileSysName](#)
[PDFFontGetCIDSystemInfo](#)
[PDFFontGetSubtype](#)
[PDTransGetSubtype](#)
[PDXObjectGetSubtype](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|---|
| <u>ASAtomExistsForString</u> | — |
| <u>ASAtomGetCount</u> | <u>ASAtomFromString</u> |

Cos conversion: None

Validity testing:

[ASAtomExistsForString](#)

ASCallback

Callbacks allow the Acrobat viewer or the Library to call functions in an application or plug-in.

Obtaining:

[ASCallbackCreate](#)

[ASCallbackCreateNotification](#)

[ASCallbackCreateProto](#)

[ASCallbackCreateReplacement](#)

Disposing:

[ASCallbackDestroy](#)

Attributes: None

Cos conversion: None

Validity testing: None

ASExtension

An opaque pointer to an object that identifies a specific plug-in. An unique *ASExtension* object is created for each plug-in when it is loaded. If the plug-in fails to initialize the *ASExtension* will remain but is marked as dead.

Obtaining:

[ASEnumExtensions](#)

Disposing: None

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|--|
| <u>ASExtensionGetFileName</u> | — |
| <u>ASExtensionGetRegisteredName</u> | — |
| — | <u>ASFileRegisterFileSys</u> <u>ASFileUnregisterFileSys</u> |
| — | <u>AVAppRegisterNotification</u> <u>AVAppUnregisterNotification</u> |
| — | <u>AVRegisterAuxDataHandler</u> <u>AVUnregisterAuxDataHandler</u> |

Cos conversion: None

Validity testing: None

ASFile

An opaque representation of an open file.

Obtaining:

[PDDocGetFile](#)
[ASFileSysOpenFile](#)
[ASFileFromMDFile](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---------------------------------------|-------------------------------|
| ASFileAcquirePathName | — |
| — | ASFileSetMode |
| ASFileGetEOF | ASFileSetEOF |
| ASFileGetFileSys | — |
| ASFileGetMDFile | — |
| ASFileGetPos | ASFileSetPos |
| — | ASFileSetMode |

Cos conversion: None

Validity testing: None

Declarations:

[ASFile Open Modes](#)
[ASFile Flags](#)
[ASFileMode Flags](#)
[ASFileStatus Flags](#)

ASFileSys

A collection of functions that implement file system services, such as opening files, deleting files, reading data from a file, and writing data to a file. Each *ASFileSys* provides these services for one class of devices.

Obtaining:

[ASGetDefaultFileSys](#)[ASFileGetFileSys](#)[ASFileGetFileSysByName](#)[PDFileSpecGetFileSys](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|------------|
| <u>ASFileGetFileSysByName</u> | — |

Cos conversion: None

Validity testing: None

Declarations:

[ASFileStatus Flags](#)[ASFileSysRec](#)

ASPathName

A file system-specific named location for a particular type of device. Uses the *ASFileSys* structure pointers for callback. An *ASPathName* is specific to a given *ASFileSys*.

Obtaining:

[ASFileAcquirePathName](#)
[ASFileSysAcquireFileSysPath](#)
[ASFileSysCreatePathName](#)
[ASFileSysPathFromDIPath](#)
[ASPathFromPlatformPath](#)
[PDFileSpecAcquireASPath](#)

Disposing:

[ASFileSysReleasePath](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|------------|
| <u>ASFileSysDIPathFromPath</u> | — |

Cos conversion: None

Validity testing: None

ASStm

A data stream that may be a buffer in memory, a file, or an arbitrary user-written procedure. Typically used to extract data from a PDF file. When writing or extracting data streams, the *ASStm* must be connected to a Cos stream.

Obtaining:

[ASFileStmRdOpen](#)

[ASMemStmRdOpen](#)

[ASProcStmRdOpen](#)

[CosStreamOpenStm](#)

Disposing:

[ASStmClose](#)

Attributes: None

Cos conversion: None

Validity testing: None

HFT

Host Function Table. The mechanism through which plug-ins call methods in the Acrobat viewer or in other plug-ins. A table of function pointers (actually callbacks).

Obtaining:

[ASExtensionMgrGetHFT](#)

Disposing:

[HFTDestroy](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|--|
| <u>HFTGetReplacedEntry</u> | <u>HFTReplaceEntry</u> |

Cos conversion: None

Validity testing:

[HFTIsValid](#)

Declarations:

[HFTs](#)

[HFTEntry](#)

[HFTEntryReplaceable](#)

HFTServer

Each *HFT* is serviced by an HFT server. The HFT server is responsible for handling requests to obtain or destroy its *HFT*.

Obtaining:

[HFTServerNew](#)

Disposing:

[HFTServerDestroy](#)

Attributes: None

Cos conversion: None

Validity testing: None

MDFFile

A file-system specific representation of an individual file. It uses the machine's native platform-specific data structure to represent a file. In the plug-in API, it is primarily used by Replacement FileSystem implementors. A replacement file system can choose it's own implementation of an *MDFFile* that is mapped by the viewer to an *ASFile* for use by clients of the replacement file system.

Obtaining:

[*ASFileGetMDFFile*](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|------------|
| <i>ASFileFromMDFFile</i> | — |

Cos conversion: None

Validity testing: None

AV Layer

Acrobat Viewer layer. A set of objects whose methods allow plug-ins to manipulate components of the Acrobat viewer application itself, such as menus and menu items.

AVActionHandler

Carries out an action. When the Acrobat viewer executes an action, it looks for the action handler with a *type* matching that of the action it is trying to execute. The Acrobat viewer invokes the matching handler to perform the action. If no match is found, the Acrobat viewer ignores the user action.

Obtaining:

[AVAppGetActionHandlerByType](#)

[AVAppEnumActionHandlers](#)

Enumerating:

[AVAppEnumActionHandlers](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|------------|
| <u>AVActionHandlerGetProcs</u> | — |
| <u>AVActionHandlerGetType</u> | — |
| <u>AVActionHandlerGetUIName</u> | — |

Cos conversion: None

Validity testing: None

Declarations:

[AVActionHandlerProcs](#)

AVAnnotHandler

Responsible for creating, displaying, selecting, and deleting a particular type of annotation. There is one annotation handler for each annotation type. The Acrobat viewer contains two built-in annotation types (notes and links), and plug-ins can add new annotation handlers by using *AVAppRegisterAnnotHandler*.

Obtaining:

[AVAppGetAnnotHandlerByName](#)

[AVAppEnumAnnotHandlers](#)

Enumerating:

[AVAppEnumAnnotHandlers](#)

Attributes: None

Cos conversion: None

Validity testing: None

Declarations:

[AVAnnotHandler](#)

AVApp

The Acrobat viewer application itself. From the application layer, you can control the appearance of Exchange, whether Exchange appears, and the size of the application window. Your application has access to the menu bar and the toolbar through this object. The application layer also provides access to the visual representation of a PDF file on the screen, that is, an *AVDoc*.

Obtaining: None

Enumerating:

[AVAppEnumActionHandlers](#)

[AVAppEnumAnnotHandlers](#)

[AVAppEnumDocs](#)

[AVAppEnumSystemFonts](#)

[AVAppEnumTools](#)

[AVAppEnumTransHandlers](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|---|
| <u>AVAppBeginModal</u> | <u>AVAppEndModal</u> |
| <u>AVAppCanQuit</u> | — |
| <u>AVAppDoingFullScreen</u> | <u>AVAppEndFullScreen</u> |
| <u>AVAppGetActionHandlerByType</u> | <u>AVAppRegisterActionHandler</u> |
| <u>AVAppGetActiveDoc</u> | — |
| <u>AVAppGetActiveTool</u> | <u>AVAppSetActiveTool</u> |
| — | <u>AVAppRegisterTool</u> |
| <u>AVAppGetAnnotHandlerByName</u> | <u>AVAppRegisterAnnotHandler</u> |
| — | <u>AVAppRegisterForPageViewAdjustCursor</u> |
| — | <u>AVAppUnregisterForPageViewAdjustCursor</u> |
| — | <u>AVAppRegisterForPageViewClicks</u> |
| — | <u>AVAppUnregisterForPageViewClicks</u> |

| <i>Get</i> | <i>Set</i> |
|---|--|
| — | <u>AVAppRegisterForPageViewDrawing</u> |
| — | <u>AVAppUnregisterForPageViewDrawing</u> |
| — | <u>AVAppRegisterIdleProc</u> |
| — | <u>AVAppUnregisterIdleProc</u> |
| — | <u>AVAppRegisterNotification</u> |
| — | <u>AVAppUnregisterNotification</u> |
| <u>AVAppGetCancelProc</u> | — |
| <u>AVAppGetDefaultTool</u> | — |
| <u>AVAppGetDocProgressMonitor</u> | — |
| — | — |
| <u>AVAppGetLanguage</u> | — |
| <u>AVAppGetLastActiveTool</u> | — |
| <u>AVAppGetMenubar</u> | — |
| <u>AVAppGetName</u> | — |
| <u>AVAppGetNumDocs</u> | — |
| <u>AVAppGetPreference</u> | <u>AVAppSetPreference</u> |
| <u>AVAppGetToolBar</u> | — |
| <u>AVAppGetToolByName</u> | — |
| <u>AVAppGetTransHandlerByType</u> | — |
| <u>AVAppGetVersion</u> | — |
| <u>AVAppHandlePlatformEvent</u> | — |
| <u>AVAppModalWindowsOpen</u> | — |
| <u>AVHasAuxDataHandler</u> | <u>AVRegisterAuxDataHandler</u> |
| — | <u>AVUnregisterAuxDataHandler</u> |

Cos conversion: None

Validity testing: None

AVDoc

A view of a PDF document in a window. There is one *AVDoc* per displayed document. Unlike a *PDDoc*, an *AVDoc* has a window associated with it.

Obtaining:

[AVAppGetActiveDoc](#)
[AVAppEnumDocs](#)
[AVDocIsReadOnly](#)
[AVDocOpenFromFile](#)
[AVDocOpenFromFileWithParams](#)
[AVDocOpenFromPDDoc](#)
[AVDocOpenFromPDDocWithParams](#)
[AVPageViewGetAVDoc](#)

Disposing:

[AVDocClose](#)

Enumerating:

[AVAppEnumDocs](#)
[AVDocEnumSelection](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|---|
| — | <u>AVDocDoActionPropsDialog</u> |
| — | <u>AVDocDoPrint</u> |
| — | <u>AVDocDoSaveAs</u> |
| — | <u>AVDocDoSaveAsWithParams</u> |
| <u>AVDocGetAVWindow</u> | — |
| <u>AVDocGetClientName</u> | <u>AVDocSetClientName</u> |
| <u>AVDocGetPageText</u> | — |

| <i>Get</i> | <i>Set</i> |
|---|--|
| AVDocGetPageView | — |
| AVDocGetPDDoc | — |
| — | AVDocClearSelection |
| — | AVDocDeleteSelection |
| — | AVDocShowSelection |
| AVDocGetSelection | AVDocSetSelection |
| AVDocGetSelectionServerByType | — |
| AVDocGetSelectionType | — |
| AVDocGetSplitterPosition | AVDocSetSplitterPosition |
| AVDocGetViewDef | AVDocSetViewDef |
| AVDocGetViewMode | AVDocSetViewMode |
| AVDocIsExternal | — |
| — | AVDocPerformAction |
| — | AVDocRegisterSelectionServer |
| — | AVDocSetDead |
| AVDocIsReadOnly | AVDocSetReadOnly |

Cos conversion: None

Validity testing: None

Declarations:

[AVDocOpenParams](#)

[AVDocPrintParams](#)

[AVDocSelectionServer](#)

[AVDocViewDef](#)

AVGrafSelect

A graphics selection on a page in a PDF file. It is a rectangular region of a page that can be copied to the clipboard as a sampled image.

Obtaining:

[AVDocGetSelection](#)

[AVGrafSelectCreate](#)

Disposing:

[AVGrafSelectDestroy](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|------------|
| <u>AVGrafSelectGetBoundingRect</u> | — |

Cos conversion: None

Validity testing: None

AVMenu

A menu in the Acrobat viewer's menu bar. Plug-ins can create new menus, add menu items at any location in any menu, and remove menu items. Deleting an *AVMenu* removes it from the menu bar (if it was attached) and deletes all the menu items it contains.

Obtaining:

[AVMenuAcquire](#)[AVMenuNew](#)[AVMenuItemAcquireSubmenu](#)[AVMenuItemGetParentMenu](#)[AVMenubarAcquireMenuByName](#)[AVMenubarAcquireMenuByIndex](#)[AVMenubarAcquireMenuByPredicate](#)

Disposing:

[AVMenuRelease](#)[AVMenuRemove](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|--|
| — | <u>AVMenuAddMenuItem</u> |
| <u>AVMenuGetMenuItemIndex</u> | — |
| <u>AVMenuGetName</u> | — |
| <u>AVMenuGetNumMenuItems</u> | — |
| <u>AVMenuGetParentMenubar</u> | — |
| <u>AVMenuGetParentMenuItem</u> | — |
| <u>AVMenuGetTitle</u> | — |

Cos conversion: None

Validity testing: None

AVMenubar

The Acrobat viewer's menu bar and a list of all menus. There is only one *AVMenubar*. Plug-ins can add new menus to or remove any menu from the menu bar. The menu bar can be hidden from the user's view.

Obtaining:

[AVAppGetMenubar](#)

[AVMenuGetParentMenubar](#)

AVAppGetMenubar is the standard way to obtain the menubar.

Attributes:

| Get | Set |
|--|---|
| — | <u>AVMenubarAddMenu</u> |
| — | <u>AVMenuRemove</u> |
| <u>AVMenubarAcquireMenuByIndex</u> | — |
| <u>AVMenubarAcquireMenuByName</u> | — |
| <u>AVMenubarAcquireMenuByPredicate</u> | — |
| <u>AVMenubarAcquireMenuItemByName</u> | — |
| <u>AVMenubarAcquireMenuItemByPredicate</u> | — |
| <u>AVMenubarGetMenuIndex</u> | — |
| <u>AVMenubarGetNumMenus</u> | — |
| <u>AVMenuItemRemove</u> | — |
| — | <u>AVMenubarHide</u> |
| — | <u>AVMenubarShow</u> |

Cos conversion: None

Validity testing: None

AVMenuItem

A menu item under a menu in the Acrobat viewer. It has a number of attributes, including a name, a keyboard shortcut, a procedure to execute when the menu item is selected, a procedure to compute whether or not the menu item is enabled, a procedure to compute whether or not the menu item is check marked, and whether or not it has a submenu.

Obtaining:

[AVMenuItemNew](#)
[AVMenuItemAcquire](#)
[AVMenubarAcquireMenuItemByName](#)
[AVMenubarAcquireMenuItemByPredicate](#)
[AVMenuAcquireMenuItemByIndex](#)
[AVMenuGetParentMenuItem](#)

Disposing:

[AVMenuItemRelease](#)
[AVMenuItemRemove](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|--|
| <u>AVMenuGetMenuItemIndex</u> | — |
| <u>AVMenuItemAcquireSubmenu</u> | — |
| <u>AVMenuItemGetLongOnly</u> | — |
| <u>AVMenuItemGetName</u> | — |
| <u>AVMenuItemGetParentMenu</u> | — |
| <u>AVMenuItemGetShortcut</u> | — |
| <u>AVMenuItemGetTitle</u> | <u>AVMenuItemSetTitle</u> |
| <u>AVMenuItemIsEnabled</u> | — |
| <u>AVMenuItemIsMarked</u> | — |
| — | <u>AVMenuItemSetComputeEnabledProc</u> |

| Get | Set |
|-----|--|
| — | AVMenuItemSetComputeMarkedProc |
| — | AVMenuItemSetExecuteProc |

Cos conversion: None

Validity testing: None

AVPageView

The area of the Acrobat viewer's window that displays the contents of a document page. Every *AVDoc* has an *AVPageView* and vice versa. It contains references to the *PDDoc* and *PDPPage* objects for the document being displayed.

Obtaining:

[AVDocGetPageView](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|---|
| <u>AVPageViewAppearanceGetAVMatrix</u> | — |
| <u>AVPageViewAcquireMachinePort</u> | <u>AVPageViewReleaseMachinePort</u> |
| <u>AVPageViewGetActiveBead</u> | — |
| <u>AVPageViewGetAnnotRect</u> | — |
| — | <u>AVPageViewSetAnnotLocation</u> |
| <u>AVPageViewGetAperture</u> | — |
| <u>AVPageViewGetAVDoc</u> | — |
| <u>AVPageViewGetColor</u> | <u>AVPageViewSetColor</u> |
| — | <u>AVPageViewShowControl</u> |
| <u>AVPageViewGetDevToPageMatrix</u> | — |
| <u>AVPageViewGetFirstVisiblePageNum</u> | — |
| <u>AVPageViewGetLastVisiblePageNum</u> | — |
| <u>AVPageViewGetLayoutMode</u> | <u>AVPageViewSetLayoutMode</u> |
| <u>AVPageViewGetMousePosition</u> | — |
| <u>AVPageViewGetNextView</u> | — |
| <u>AVPageViewGetPage</u> | — |
| <u>AVPageViewGetPageNum</u> | <u>AVPageViewSetPageNum</u> |
| — | <u>AVPageViewGoBack</u> |
| — | <u>AVPageViewGoForward</u> |

| Get | Set |
|---|--|
| — | AVPageViewGoTo |
| AVPageViewGetPageToDevMatrix | — |
| AVPageViewGetSelectedAnnotPageNum | — |
| AVPageViewGetThreadIndex | — |
| AVPageViewGetVisibleAnnotPage | — |
| AVPageViewGetZoom | AVPageViewZoomTo |
| AVPageViewGetZoomType | — |
| — | AVPageViewDeviceRectToPageRZ |
| — | AVPageViewDrawCosObj |
| — | AVPageViewDragRect |
| — | AVPageViewDrawNow |
| — | AVPageViewDrawRect |
| — | AVPageViewDrawRectOutline |
| — | AVPageViewHighlightText |
| — | AVPageViewInsetRect |
| — | AVPageViewInvalidateRect |
| — | AVPageViewInvalidateText |
| — | AVPageViewInvertRect |
| — | AVPageViewInvertRectOutline |
| AVPageViewInvertQuad | — |
| AVPageViewIsBeadAtPoint | — |
| AVPageViewPageNumIsVisible | — |
| AVPageViewPointInText | — |
| — | AVPageViewReadPageDown |
| — | AVPageViewReadPageUp |
| — | AVPageViewScrollTo |
| — | AVPageViewScrollToRect |
| — | AVPageViewSetAnnotLocation |

| <i>Get</i> | <i>Set</i> |
|---|---|
| <u>AVPageViewToViewDest</u> | <u>AVPageViewUseThisDestination</u> |

Cos conversion: None

Validity testing: None

AVSys

Provides various system-wide utilities, including setting the cursor shape and beeping.

Obtaining: None

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|---------------------------------------|
| <u>AVSysGetCursor</u> | <u>AVSysSetCursor</u> |
| <u>AVSysGetModifiers</u> | — |
| <u>AVSysGetStandardCursor</u> | — |
| <u>AVSysMouseIsStillDown</u> | — |

Cos conversion: None

Validity testing: None

AVTool

Handles key presses and mouse clicks in the content region of an *AVPageView*. *AVTools* do not handle mouse clicks in other parts of the viewer's window, such as in the bookmark pane. At any time, there is one active tool.

Obtaining:

[AVAppGetActiveTool](#)[AVAppGetLastActiveTool](#)[AVAppGetDefaultTool](#)[AVAppGetToolByName](#)[AVAppEnumTools](#)

Enumerating:

[AVAppEnumTools](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|------------|
| <u>AVToolGetType</u> | — |
| <u>AVToolsPersistent</u> | — |

Cos conversion: None

Validity testing: None

Declarations:

[AVTool](#)

AVToolBar

The Acrobat viewer's toolbar (the palette of buttons). There is only one *AVToolBar*.

Obtaining:

[AVAppGetToolBar](#)

[AVToolBarNewFlyout](#)

[AVToolButtonGetFlyout](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|--|
| <u>AVToolBarGetButtonByName</u> | — |
| <u>AVToolBarGetFrame</u> | — |
| <u>AVToolBarGetNumButtons</u> | — |
| <u>AVToolBarIsRoomFor</u> | — |
| — | <u>AVToolBarAddButton</u> |
| — | <u>AVToolBarUpdateButtonStates</u> |
| — | <u>AVToolButtonDestroy</u> |
| — | <u>AVToolButtonRemove</u> |
| — | <u>AVToolButtonSetExternal</u> |

Cos conversion: None

Validity testing: None

AVToolButton

A button in the Acrobat viewer's toolbar. Like menu items, the procedure that executes when the button is clicked can be set by a plug-in. Although not required, there is generally a menu item corresponding to each button, allowing users to select a function using either the button or the menu item. Buttons are added to the toolbar by specifying which existing button they appear before or after.

Obtaining:

[AVToolBarGetButtonByName](#)[AVToolButtonNew](#)[AVToolBarEnumButtons](#)

Disposing:

[AVToolButtonDestroy](#)[AVToolButtonRemove](#)

Enumerating:

[AVToolBarEnumButtons](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|--|
| <u>AVToolButtonGetFlyout</u> | <u>AVToolButtonSetFlyout</u> |
| <u>AVToolButtonGetIcon</u> | <u>AVToolButtonSetIcon</u> |
| <u>AVToolButtonGetMenu</u> | <u>AVToolButtonSetMenu</u> |
| <u>AVToolButtonIsEnabled</u> | — |
| <u>AVToolButtonIsMarked</u> | — |
| <u>AVToolButtonIsSeparator</u> | — |
| — | <u>AVToolButtonSetComputeEnabledProc</u> |
| — | <u>AVToolButtonSetComputeMarkedProc</u> |
| — | <u>AVToolButtonSetExecuteProc</u> |

| <i>Get</i> | <i>Set</i> |
|------------|---|
| — | AVToolButtonSetExternal |
| — | AVToolButtonSetHelpText |

Cos conversion: None

Validity testing: None

Declarations:

[Tool Button Flags](#)

AVWindow

Creates and manages windows. Plug-ins should use *AVWindows* for their own dialogs, floating palettes, and so forth, to ensure that those windows work well with the Acrobat viewer. For example, under Windows they are hidden when the Acrobat viewer is iconified. Once the plug-in creates an *AVWindow*, it is free to use platform-dependent code to put whatever it wants in the window.

Obtaining:

[AVWindowNew](#)
[AVWindowNewFromPlatformThing](#)

[AVDocGetAVWindow](#)

Disposing:

[AVWindowDestroy](#)

[AVWindowUserClose](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|---|
| — | <u>AVWindowDrawNow</u> |
| <u>AVWindowGetFrame</u> | <u>AVWindowSetFrame</u> |
| <u>AVWindowGetInterior</u> | — |
| <u>AVWindowGetOwnerData</u> | <u>AVWindowSetOwnerData</u> |
| <u>AVWindowGetPlatformThing</u> | — |
| <u>AVWindowGetTitle</u> | <u>AVWindowSetTitle</u> |
| <u>AVWindowIsKey</u> | <u>AVWindowBecomeKey</u> |
| — | <u>AVWindowSetWantsKey</u> |
| — | <u>AVWindowResignKey</u> |
| — | <u>AVWindowBringToFront</u> |
| <u>AVWindowIsVisible</u> | <u>AVWindowShow</u> |
| — | <u>AVWindowHide</u> |
| — | <u>AVWindowInvalidateRect</u> |

| Get | Set |
|-----|----------------------------------|
| — | AVWindowMaximize |

Cos conversion: None

Validity testing: None

Declarations:

| |
|---------------------------------|
| AVWindow Flags |
| AVWindowHandler |
| AVWindowLayer |

Cos Layer

A set of objects that provide access to the building blocks used to construct documents. Its methods allow applications to manipulate the low-end data in a PDF file, such as strings, numbers, and dictionaries.

CosDoc

A Cos level representation of an entire PDF file.

Obtaining:

[CosDocCreate](#)
[CosDocOpenWithParams](#)
[CosObjGetDoc](#)
[PDDocGetCosDoc](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|-----------------------------------|--------------------------------------|
| — | CosDocClose |
| CosDocGetInfoDict | — |
| CosDocGetObjByID | — |
| — | CosDocSaveToFile |
| — | CosDocSaveWithParams |

Cos conversion: See the list of “Obtaining” methods.

Validity testing: None

Declarations:

[CosDocCreate Flags](#)
[CosDocSave Flags](#)
[CosDocSaveParams](#)

CosObj

A general object in a PDF file, which may be of any Cos object type.

Obtaining:

[CosArrayGet](#)

[CosDictGet](#)

[CosDocGetInfoDict](#)

[CosDocGetObjByID](#)

[CosNewBoolean](#)

[CosNewDict](#)

[CosNewFixed](#)

[CosNewInteger](#)

[CosNewName](#)

[CosNewNull](#)

[CosNewStream](#)

[CosNewString](#)

[CosStreamDict](#)

[PDActionGetCosObj](#)

[PDAnnotGetCosObj](#)

[PDBeadGetCosObj](#)

[PDBookmarkGetCosObj](#)

[PDCharProcGetCosObj](#)

[PDFileSpecGetCosObj](#)

[PDFontGetCosObj](#)

[PDFormGetXUIDCosObj](#)

[PDImageSelectAlternate](#)

[PDNameTreeGetCosObj](#)

[PDPageGetCosObj](#)

[PDPageGetCosResources](#)

[PDPageLabelGetCosObj](#)
[PDThreadGetCosObj](#)
[PDTransGetCosObj](#)
[PDViewDestGetCosObj](#)
[PDXObjectGetCosObj](#)

Disposing:

[CosObjDestroy](#)

Enumerating:

[CosObjEnum](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|--|
| <u>CosArrayGet</u> | <u>CosArrayInsert</u> |
| <u>CosArrayLength</u> | <u>CosArrayPut</u> |
| <u>CosArrayRemoveNth</u> | |
| — | <u>CosArrayRemove</u> |
| <u>CosBooleanValue</u> | — |
| <u>CosDictGet</u> | — |
| <u>CosDictKnown</u> | <u>CosDictPut</u> |
| — | <u>CosDictRemove</u> |
| <u>CosDocGetObjByID</u> | — |
| <u>CosFixedValue</u> | — |
| <u>CosIntegerValue</u> | — |
| <u>CosNameValue</u> | — |
| <u>CosObjEqual</u> | — |
| <u>CosObjGetDoc</u> | — |
| <u>CosObjGetGeneration</u> | — |
| <u>CosStringGetHexFlag</u> | <u>CosStringSetHexFlag</u> |

| <i>Get</i> | <i>Set</i> |
|----------------------------------|------------|
| CosObjGetID | — |
| CosDocGetObjByID | — |
| CosObjHash | — |
| CosObjIsIndirect | — |
| CosStreamDict | — |
| CosStreamLength | — |
| CosStreamPos | — |
| CosStringValue | — |

Cos conversion: See “Cos conversion” for each object

Validity testing: None

Declarations:

[Cos Object Types](#)

PD Layer

A group of objects that provide access to components of PDF documents such as pages, annotations, and fonts. Its methods allow applications to manipulate document components.

PDAction

Actions are what happens when a user clicks on a link or bookmark. In addition, the Acrobat viewer allows a document to have an action that is executed automatically when the document is opened. Applications can also support actions in custom annotation types they add.

Obtaining:

[PDActionNew](#)
[PDActionNewFromDest](#)
[PDActionNewFromFileSpec](#)
[PDLinkAnnotGetAction](#)
[PDBookmarkGetAction](#)
[PDDocGetOpenAction](#)

Disposing:

[PDActionDestroy](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|------------|
| <u>PDActionEqual</u> | — |
| <u>PDActionGetDest</u> | — |
| <u>PDActionGetFileSpec</u> | — |
| <u>PDActionGetSubtype</u> | — |

Cos conversion:

[PDActionGetCosObj](#)
[PDActionFromCosObj](#)

Validity testing:

[PDActionIsValid](#)

PDAnnot

An annotation on a page in a PDF file. Acrobat viewers have two built-in annotation types: *PDTextAnnot* and *PDLinkAnnot*. Physical attributes of the annotation can be set and queried. Plug-ins add movie and Widget (form field) annotations. Developers can define new annotation subtypes by creating new annotation handlers.

Obtaining:

[AVPageViewInvertQuad](#)

[PDAnnotFromCosObj](#)

[PDPageAddNewAnnot](#)

[PDPageCreateAnnot](#)

[PDPageGetAnnot](#)

Disposing:

[PDPageRemoveAnnot](#)

Attributes:

| Get | Set |
|--|---|
| <u>AVPageViewGetSelectedAnnotPageNum</u> | — |
| — | <u>AVPageViewSetAnnotLocation</u> |
| <u>PDAnnotEqual</u> | — |
| <u>PDAnnotGetColor</u> | <u>PDAnnotSetColor</u> |
| <u>PDAnnotGetDate</u> | <u>PDAnnotSetDate</u> |
| <u>PDAnnotGetFlags</u> | <u>PDAnnotSetFlags</u> |
| <u>PDAnnotGetRect</u> | <u>PDAnnotSetRect</u> |
| <u>PDAnnotGetSubtype</u> | — |
| <u>PDAnnotGetTitle</u> | <u>PDAnnotSetTitle</u> |

Cos conversion:

[PDAnnotGetCosObj](#)

[PDAnnotFromCosObj](#)

Validity testing:

[PDAnnotIsValid](#)

Declarations:

[PDAnnot Flags](#)

PDBead

A single rectangle in an article thread. (Article threads are known simply as articles in the Acrobat viewer's user interface.) A bead remains valid as long as a thread is "current and active."

Obtaining:

[AVPageViewGetActiveBead](#)

[PDThreadGetFirstBead](#)

[PDBeadNew](#)

[PDBeadGetNext](#)

[PDBeadGetPrev](#)

[PDBeadFromCosObj](#)

Disposing:

[PDBeadDestroy](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|--------------------------------------|
| <u>PDBeadGetIndex</u> | — |
| <u>PDBeadGetNext</u> | — |
| <u>PDBeadGetPrev</u> | — |
| <u>PDBeadGetRect</u> | <u>PDBeadSetRect</u> |
| <u>PDBeadGetThread</u> | — |
| <u>PDBeadIsValid</u> | — |
| <u>PDBeadSetPage</u> | — |

Cos conversion:

[PDBeadGetCosObj](#)

[PDBeadFromCosObj](#)

Validity testing:

PDBeadIsValid

PDBookmark

A bookmark on a page in a PDF file. Each bookmark has a title that appears on screen, and an action that specifies what happens when a user clicks on the bookmark. Bookmarks can either be created interactively by the user through the Acrobat viewer's user interface or programmatically generated. The typical action for a user-created bookmark is to move to another location in the current document, although any action (see [PDAction](#)) can be specified.

Obtaining:

[PDDocExportNotes](#)
[PDBookmarkAddNewSibling](#)
[PDBookmarkAddNewChild](#)
[PDBookmarkFromCosObj](#)
[PDBookmarkGetByTitle](#)
[PDBookmarkGetParent](#)
[PDBookmarkGetFirstChild](#)
[PDBookmarkGetLastChild](#)
[PDBookmarkGetNext](#)
[PDBookmarkGetPrev](#)

Disposing:

[PDBookmarkDestroy](#)
[PDBookmarkUnlink](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|---|
| PDBookmarkEqual | — |
| PDBookmarkGetAction | PDBookmarkSetAction |
| PDBookmarkGetCount | — |
| PDBookmarkGetFirstChild | PDBookmarkAddChild , PDBookmarkAddNewChild |

| Get | Set |
|--|--|
| PDBookmarkGetIndent | — |
| PDBookmarkGetLastChild | PDBookmarkAddChild , PDBookmarkAddNewChild |
| PDBookmarkGetNext | PDBookmarkAddNext , PDBookmarkAddNewSibling |
| PDBookmarkGetParent | PDBookmarkAddSubtree |
| PDBookmarkGetPrev | PDBookmarkAddPrev , PDBookmarkAddNewSibling |
| PDBookmarkGetTitle | PDBookmarkSetTitle |
| PDBookmarkHasChildren | — |
| PDBookmarkIsOpen | PDBookmarkSetOpen |

Cos conversion:

| |
|--------------------------------------|
| PDBookmarkGetCosObj |
| PDBookmarkFromCosObj |

Validity testing:

| |
|-----------------------------------|
| PDBookmarkIsValid |
|-----------------------------------|

PDCharProc

A character procedure, a stream of graphic operators (see [PDGraphic](#)) that draw a particular glyph of a Type 3 PostScript font.

Obtaining:

[PDFontEnumCharProcs](#)

Enumerating:

[PDCharProcEnum](#)

[PDFontEnumCharProcs](#)

Attributes: None

Cos conversion:

[PDCharProcGetCosObj](#)

Validity testing: None

PDDoc

The underlying PDF representation of a document. There is a correspondence between a *PDDoc* and an *ASFile*; the *PDDoc* object is the hidden object behind every *AVDoc*. An *ASFile* may have zero or more underlying files, so a PDF file does not always correspond to a single disk file. For example, an *ASFile* may provide access to PDF data in a database.

Through *PDDocs*, your application can perform most of the Edit | Pages menu items from Exchange (delete, replace, and so on). Thumbnails can be created and deleted through this object. You can set and retrieve document information fields through this object as well. The first page in a *PDDoc* is page 0.

Obtaining:

[AVDocGetPDDoc](#)
[PDDocFromCosDoc](#)
[PDDocOpen](#)
[PDDocOpenFromASFile](#)
[PDDocOpenWithParams](#)
[PDDocCreate](#)
[PDPAGEGetDoc](#)
[PDFFileSpecGetDoc](#)
[PDEnumDocs](#)

Disposing:

[PDDocClose](#)
[PDDocRelease](#)

Enumerating:

[PDDocEnumFonts](#)
[PDDocEnumLoadedFonts](#)
[PDEnumDocs](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|---|
| AVAuthOpen | — |
| — | PDDocAcquire |
| PDDocAcquirePage | PDDocCreatePage |
| | PDDocDeletePages |
| | PDDocImportNotes |
| | PDDocMovePage |
| | PDDocReplacePages |
| — | PDDocAuthorize |
| PDDocExportNotes | — |
| PDDocGetCryptHandlerClientData | — |
| PDDocGetFile | — |
| PDDocGetFlags | PDDocClearFlags |
| | PDDocSetFlags |
| PDDocGetFullScreen | PDDocSetFullScreen |
| PDDocGetID | — |
| PDDocGetInfo | PDDocSetInfo |
| PDDocGetLabelForPageNum | PDDocRemovePageLabel |
| PDDocGetNameTree | PDDocRemoveNameTree |
| PDDocGetNewCryptHandler | PDDocSetNewCryptHandler |
| PDDocGetNewSecurityData | PDDocNewSecurityData |
| | PDDocSetNewSecurityData |
| PDDocGetNewSecurityInfo | — |
| PDDocGetNumPages | — |
| PDDocGetNumThreads | — |
| PDDocGetOpenAction | PDDocSetOpenAction |
| PDDocGetPageLabel | PDDocSetPageLabel |
| PDDocFindPageNumForLabel | — |
| PDDocGetPermissions | PDDocAuthorize |
| PDDocGetSecurityData | — |

| Get | Set |
|--|---|
| PDDocGetStructTreeRoot | PDDocRemoveStructTreeRoot |
| | PDDocCreateStructTreeRoot |
| PDDocGetThread | PDDocAddThread |
| | PDDocRemovePageLabel |
| PDDocGetThreadIndex | — |
| — | PDDocCreateThumbs |
| | PDDocDeleteThumbs |
| PDDocGetWordFinder | PDDocCreateWordFinder |
| | PDDocCreateWordFinderUCS |
| | PDWordFinderDestroy |
| PDDocGetVersion | — |
| — | PDDocSave |
| | PDDocSaveWithParams |

Cos Conversion:

[PDDocGetCosDoc](#)

Validity testing: None

Declarations:

[PDDocFlags](#)

[PDDocReadAhead Flags](#)

[PDDocSaveParams](#)

PDFFileSpec

The PDF file specification object. It is used to specify a file in an action (see [PDAction](#)). A file specification in a PDF file can take two forms:

- A single platform-independent pathname.
- A data structure containing one or more alternative ways to locate the file on different platforms.

PDFFileSpecs can be created from *ASPathNames* or from Cos objects.

Obtaining:

[PDActionGetFileSpec](#)[PDFFileSpecNewFromASPath](#)[PDFFileSpecFromCosObj](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|------------|
| PDFFileSpecAcquireASPath | — |
| PDFFileSpecGetDIPath | — |
| PDFFileSpecGetDoc | — |
| PDFFileSpecGetFileSys | — |
| PDFFileSpecGetFileSysName | — |

Cos conversion:

[PDFFileSpecGetCosObj](#)[PDFFileSpecFromCosObj](#)

Validity testing:

[PDFFileSpecsValid](#)

Declarations:

[PDFFileSpecHandler](#)

PDFont

A font that is used to draw text on a page. It corresponds to a Font Resource in a PDF file. Applications can get a list of *PDFonts* used on a *PDPage* or a range of *PDPages*. More than one *PDPage* may reference the same *PDFont* object.

A *PDFont* has a number of attributes whose values can be read or set, including an array of widths, the character encoding, and the font's resource name.

Obtaining:

[PDDocEnumFonts](#)
[PDDocEnumLoadedFonts](#)
[PDFontGetDescendant](#)
[PDStyleGetFont](#)

Enumerating:

[PDDocEnumFonts](#)
[PDFontEnumCharProcs](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|--|
| PDFontAcquireEncodingArray | PDFontEncodingArrayRelease |
| PDFontAcquireXlateTable | PDFontXlateTableRelease |
| PDFontGetBBox | — |
| PDFontGetCharSet | — |
| PDFontGetCIDSystemInfo | — |
| PDFontGetCIDSystemSupplement | — |
| PDFontGetDescendant | — |
| PDFontGetEncodingIndex | — |
| PDFontGetEncodingName | — |
| PDFontGetFontMatrix | — |

| Get | Set |
|----------------------------------|----------------------------------|
| PDFontGetMetrics | PDFontSetMetrics |
| PDFontGetName | — |
| PDFontGetSubtype | — |
| PDFontGetWidths | — |
| PDFontIsEmbedded | — |

Cos conversion:

[PDFontGetCosObj](#)

Validity testing: None

PDFForm

A self-contained set of graphics operators (essentially a subroutine of PDF page-marking operators) that is used when a particular graphic is drawn more than once in the document. It corresponds to a Form resource.

Obtaining: None

Enumerating:

[PDFFormEnumPaintProc](#)

[PDFFormEnumResources](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--------------------------------------|------------|
| PDFFormGetBBox | — |
| PDFFormGetFormType | — |
| PDFFormGetMatrix | — |
| PDFFormGetXUIDCosObj | — |

Cos Conversion:

[PDXObjectGetCosObj](#)

Validity testing: None

PDGraphic

All graphic objects that comprise page, charproc, and *PDForm* descriptions.

Obtaining: None

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|------------|
| PDGraphicGetBBox | — |
| PDGraphicGetCurrentMatrix | — |
| PDGraphicGetState | — |

Cos Conversion: None

Validity testing: None

Declarations:

[PDGraphicEnumMonitor](#)

PDImage

A sampled image or image mask that corresponds to a PDF Image resource. You can use any *PDXObject* method on a *PDImage*.

Obtaining: None

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|------------|
| PDImageColorSpaceGetIndexLookup | — |
| PDImageGetAttrs | — |

Cos Conversion:

| |
|------------------------------------|
| PDXObjectGetCosObj |
|------------------------------------|

Validity testing: None

PDInlineImage

An image whose data is stored in the page description's contents stream—instead of being stored as an image resource (see [PDImage](#)).

Obtaining: None

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|------------|
| PDInlineImageColorSpaceGetIndexLookup | — |
| PDInlineImageGetAttrs | — |
| PDInlineImageGetData | — |

Cos conversion: None

Validity testing: None

PDLinkAnnot

A link annotation on a page in a PDF file. You can use any *PDAnnot* method on a *PDLinkAnnot*. Applications can:

- Get and set the bounding rectangle and color, using *PDAnnot* methods.
- Get and set the action that occurs when the link is activated, and the link's border, using *PDLinkAnnot* methods.
- Create new link annotations and delete existing ones, using the *PDPage* methods.

Obtaining:

Any of the [PDAnnot](#) calls, followed by [CastToPDLinkAnnot](#). The annotation passed to [CastToPDLinkAnnot](#) must be a link annotation, it will not convert other annotation types into link annotations.

Disposing:

[PDPageRemoveAnnot](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|--|
| PDLinkAnnotGetAction | PDLinkAnnotSetAction |
| PDLinkAnnotGetBorder | PDLinkAnnotSetBorder |
| AVPageViewGetSelectedAnnotPageNum | — |
| — | AVPageViewSetAnnotLocation |
| PDAnnotEqual | — |
| PDAnnotGetColor | PDAnnotSetColor |
| PDAnnotGetDate | PDAnnotSetDate |
| PDAnnotGetFlags | PDAnnotSetFlags |

| Get | Set |
|-----------------------------------|---------------------------------|
| PDAnnotGetRect | PDAnnotSetRect |
| PDAnnotGetSubtype | — |
| PDAnnotGetTitle | PDAnnotSetTitle |

Cos conversion:

| |
|-----------------------------------|
| PDAnnotGetCosObj |
| PDAnnotFromCosObj |

Validity testing:

| |
|--------------------------------|
| PDAnnotIsValid |
|--------------------------------|

Declarations:

| |
|-----------------------------------|
| PDLinkAnnotBorder |
|-----------------------------------|

PDNameTree

The dictionary used to store all of the Named Destinations in a PDF file. A name tree is used to map Cos strings to Cos objects just as a Cos dictionary is used to map Cos names to Cos objects. However, a name tree can have many more entries than a Cos dictionary can. You create a *PDNameTree* and locate it where you think is appropriate (perhaps under a page, but most often right under the catalog).

Name trees use Cos-style strings (not *NULL*-terminated C strings), which may use Unicode encoding, and these may contain bytes with zeroes in them (high bytes of ASCII characters).

Obtaining:

[PDDocCreateNameTree](#)[PDNameTreeLookup](#)[PDNameTreeNew](#)[PDNameTreeFromCosObj](#)

Disposing: None

Enumerating:

[PDNameTreeEnum](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|---|
| <u>PDDocGetNameTree</u> | — |
| <u>PDNameTreeEqual</u> | — |
| <u>PDNameTreeGet</u> | <u>PDNameTreePut</u> |
| — | <u>PDNameTreeRemove</u> |

Cos conversion:

[PDNameTreeGetCosObj](#)

Validity testing:

PDNameTreelsValid

PDNumTree

An object that points to the root node of a number tree inside a PDF file. A number tree is used to map integers to arbitrary Cos objects just as a Cos dictionary is used to map Cos names to Cos objects. However, a number tree can have many more entries than a Cos dictionary can.

Obtaining:

[PDNumTreeNew](#)

[PDNumTreeFromCosObj](#)

Disposing: None

Enumerating:

[PDNumTreeEnum](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---------------------------------------|--|
| <u>PDNumTreeEqual</u> | — |
| <u>PDNumTreeGet</u> | <u>PDNumTreePut</u> |
| — | <u>PDNumTreeRemove</u> |

Cos conversion:

[PDNumTreeGetCosObj](#)

Validity testing:

[PDNumTreeIsValid](#)

PDPage

A single page in the PDF representation of a document. Just as PDF files are partially composed of their pages, *PDDocs* are composed of *PDPages*. A page contains a series of objects representing the objects drawn on the page (*PDGraphic*), a list of resources used in drawing the page, annotations (*PDAnnot*), an optional thumbnail image of the page, and the beads used in any articles that occur on the page. The first page in a *PDDoc* is page 0.

Obtaining:

[PDDocCreatePage](#)
[PDBeadAcquirePage](#)
[PDDocAcquirePage](#)
[AVPageViewGetPage](#)

Disposing:

[PDDocDeletePages](#)
[PDPageRelease](#)

Enumerating: None

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|--|
| — | PDDocAcquirePage |
| PDPageEnumResources | PDPageAddAnnot |
| PDPageGetAnnotIndex | PDPageAddNewAnnot |
| PDPageGetAnnotSequence | PDPageCreateAnnot |
| | PDPageRemoveAnnot |
| PDPageGetBBox | — |
| PDPageGetCosResources | PDPageAddCosResource |
| — | PDPageAddCosContents |
| | PDPageDrawContentsToWindow |
| | PDPageRemoveCosContents |
| | PDPageRemoveCosResource |

| <i>Get</i> | <i>Set</i> |
|--|-------------------------------------|
| PDPageGetCropBox | PDPageSetCropBox |
| PDPageGetDefaultMatrix | — |
| PDPageGetDoc | — |
| PDPageGetDuration | PDPageSetDuration |
| PDPageGetFlippedMatrix | — |
| PDPageGetMediaBox | PDPageSetMediaBox |
| PDPageGetNumAnnots | — |
| PDPageGetNumber | — |
| PDPageGetRotate | PDPageSetRotate |
| PDPageGetTransition | PDPageSetTransition |
| PDPageHasTransition | — |

Cos conversion:

[PDPageGetCosObj](#)

Validity testing: None

Declarations:

[PDPageMode](#)

PDPageLabel

A label used to describe a page. This is used to allow for non-sequential page numbering or the addition of arbitrary labels for a page (such as the inclusion of Roman numerals at the beginning of a book). A *PDPageLabel* specifies the numbering style to use (for example, upper- or lower-case Roman, decimal, and so forth), the starting number for the first page, and an arbitrary prefix to be prepended to each number (for example, "A-" to generate "A-1", "A-2", "A-3", and so forth.)

Obtaining:

[PDDocGetPageLabel](#)
[PDDocGetLabelForPageNum](#)
[PDPageLabelFromCosObj](#)
[PDPageLabelNew](#)

Disposing:

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|-----------------------------------|
| PDPageLabelEqual | — |
| — | PDDocSetPageLabel |
| PDPageLabelGetStyle | — |
| PDPageLabelGetPrefix | — |
| PDPageLabelGetStart | — |
| PDDocFindPageNumForLabel | — |

Cos conversion:

[PDPageLabelGetCosObj](#)

Validity testing:

PDPageLabellsValid

PDPath

A *PDPath* is a graphic object representing a path in a page description. Paths are arbitrary shapes made of straight lines, rectangles, and cubic curves. Path objects can be stroked, filled, and/or serve as a clipping path.

Obtaining: None

Enumerating:

[PDPathEnum](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|----------------------------------|------------|
| PDPathGetPaintOp | — |

Cos conversion: None

Validity testing: None

Declarations:

[PDPathEnumMonitor](#)

[PDPathPaintOp](#)

PDStyle

Provides access to information about the fonts, font sizes, and colors used in a *PDWord*.

Obtaining:

[PDWordGetNthCharStyle](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|------------|
| <u>PDStyleGetColor</u> | — |
| <u>PDStyleGetFont</u> | — |
| <u>PDStyleGetFontSize</u> | — |

Cos conversion: None

Validity testing: None

PDText

A graphic object representing one or more character strings on a page in a PDF file. Like paths, text can be stroked, filled, and/or serve as a clipping path.

Obtaining: None

Enumerating:

[PDTextEnum](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--------------------------------|------------|
| PDTextGetState | — |

Cos conversion: None

Validity testing: None

PDTextAnnot

A PDF text annotation on a page in a PDF file. You can use any *PDAnnot* method on a *PDTextAnnot*.

Applications can:

- Get and set attributes including the rectangle, textual contents, and whether or not the annotation is open.
- Create new text annotations and delete or move existing ones using *PDAnnot* methods.
- Manipulate the behavior of text annotations by modifying the Text Annotation Handler.

Obtaining:

Any of the [PDAnnot](#) calls, followed by [CastToPDTextAnnot](#). The annotation passed to [CastToPDTextAnnot](#) must be a text annotation, it will not convert other annotation types into text annotations.

Disposing:

[PDPageRemoveAnnot](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|--|
| PDTextAnnotGetContents | PDTextAnnotSetContents |
| PDTextAnnotIsOpen | PDTextAnnotSetOpen |
| AVPageViewGetSelectedAnnotPageNum | — |
| — | AVPageViewSetAnnotLocation |
| PDAnnotEqual | — |
| PDAnnotGetColor | PDAnnotSetColor |
| PDAnnotGetDate | PDAnnotSetDate |

| <i>Get</i> | <i>Set</i> |
|--|--|
| <u>PDAnnotGetFlags</u> | <u>PDAnnotSetFlags</u> |
| <u>PDAnnotGetRect</u> | <u>PDAnnotSetRect</u> |
| <u>PDAnnotGetSubtype</u> | — |
| <u>PDAnnotGetTitle</u> | <u>PDAnnotSetTitle</u> |

Cos conversion:

| |
|--|
| <u>PDAnnotGetCosObj</u> |
| <u>PDAnnotFromCosObj</u> |

Validity testing:

| |
|---------------------------------------|
| <u>PDAnnotIsValid</u> |
|---------------------------------------|

PDTextSelect

A selection of text on a single page, and may contain more than one disjoint group of words. A text selection is specified by one or more *ranges* of text, with each range containing the word numbers of the selected words. Each range specifies a start and end word, where “start” is the first of a series of selected words and “end” is the first word *not* in the series.

Obtaining:

[AVDocGetSelection](#)[AVPageViewTrackText](#)[PDDocCreateStructTreeRoot](#)[PDTextSelectCreatePageHilite](#)[PDTextSelectCreateWordHilite](#)[PDTextSelectCreateRanges](#)

Disposing:

[PDTextSelectDestroy](#)

Enumerating:

[PDTextSelectEnumQuads](#)[PDTextSelectEnumText](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|---|
| — | <u>PDTextSelectCreatePageHilite</u> |
| — | <u>PDTextSelectCreateRanges</u> |
| — | <u>PDTextSelectCreateWordHilite</u> |
| <u>PDTextSelectGetBoundingRect</u> | <u>PDTextSelectGetBoundingRect</u> |
| <u>PDTextSelectGetPage</u> | — |
| <u>PDTextSelectGetRange</u> | — |
| <u>PDTextSelectGetRangeCount</u> | — |

Cos conversion: None

Validity testing: None

Declarations:

[PDTextSelectRange](#)

PDThread

An article in the Acrobat viewer's user interface, and contains an ordered sequence of rectangles that bound the article. Each rectangle is called a *bead*. Threads can be created interactively by the user, or programmatically.

Obtaining:

[PDDocGetThread](#)

[PDThreadNew](#)

[PDThreadFromCosObj](#)

[PDBeadGetThread](#)

Disposing:

[PDDocRemovePageLabel](#)

[PDThreadDestroy](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|---|
| — | <u>PDThreadDestroy</u> |
| <u>PDThreadGetFirstBead</u> | <u>PDThreadSetFirstBead</u> |
| <u>PDThreadGetInfo</u> | <u>PDThreadSetInfo</u> |
| — | <u>PDBeadInsert</u> |

Cos conversion:

[PDThreadGetCosObj](#)

[PDThreadFromCosObj](#)

Validity testing:

[PDThreadIsValid](#)

PDThumb

A thumbnail preview image of a page.

Obtaining: None

Disposing: None

Attributes: None

Cos conversion: None

Validity testing: None

PDTrans

A transition to a page. The **Trans** key in a Page dictionary specifies a Transition dictionary, which describes the effect to use when going to a page and the amount of time the transition should take.

Obtaining:

[PDPageGetTransition](#)

[PDTransFromCosObj](#)

[PDTransNew](#)

[PDTransNewFromCosDoc](#)

[PDTransNull](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|------------|
| <u>PDTransEqual</u> | — |
| <u>PDTransGetDuration</u> | — |
| <u>PDTransGetSubtype</u> | — |

Cos conversion:

[PDTransFromCosObj](#)

[PDTransGetCosObj](#)

Validity testing:

[PDTransIsValid](#)

Declarations:

[Transition Duration](#)

PDViewDestination

A particular view of a page in a document. It contains a reference to a page, a rectangle on that page, and information specifying how to adjust the view to fit the window's size and shape. It corresponds to a PDF Dest array and can be considered a special form of a *PDAction*.

Obtaining:

[AVPageViewToViewDest](#)[PDActionGetDest](#)[PDViewDestCreate](#)[PDViewDestFromCosObj](#)[PDViewDestResolve](#)

Disposing:

[PDViewDestDestroy](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|------------|
| <u>PDViewDestGetAttr</u> | — |

Cos Conversion:

[PDViewDestFromCosObj](#)[PDViewDestGetCosObj](#)

Validity testing:

[PDViewDestIsValid](#)

PDWord

A word in a PDF file. Each word contains a sequence of characters in one or more styles (see [PDStyle](#)).

Obtaining:

[PDWordFinderGetNthWord](#)

[PDWordFinderEnumWords](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|------------|
| PDWordGetAttr | — |
| PDWordGetCharacterTypes | — |
| PDWordGetCharDelta | — |
| PDWordGetCharOffset | — |
| PDWordGetLength | — |
| PDWordGetNthCharStyle | — |
| PDWordGetNthQuad | — |
| PDWordGetNumQuads | — |
| PDWordGetString | — |
| PDWordGetStyleTransition | — |
| PDWordIsRotated | — |

Cos Conversion: None

Validity testing: None

Declarations:

[Word Attributes](#)

PDWordFinder

Extracts words from a PDF file, and enumerates the words on a single page or on all pages in a document.

Obtaining:

[PDDocCreateWordFinder](#)

[PDDocCreateWordFinderUCS](#)

[PDDocGetWordFinder](#)

Disposing:

[PDWordFinderDestroy](#)

Enumerating:

[PDWordFinderEnumWords](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|---|
| PDWordFinderAcquireWordList | PDWordFinderReleaseWordList |
| PDWordFinderGetLatestAlgVersion | — |
| PDWordFinderGetNthWord | — |

Cos Conversion: None

Validity testing: None

Declarations:

[Word Finder Sort Order Flags](#)

PDXObject

A PDF *XObject*. There are no *PDXObject* objects used directly in the Acrobat API, but Acrobat viewers currently use two *XObject* subclasses: [PDEImage](#) and [PDEForm](#), and a third *XObject* type: *procset*. You can use any *PDXObject* method on these three objects.

Obtaining: None

Enumerating:

[PDXObjectEnumFilters](#)

[PDXObjectGetData](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|------------|
| PDXObjectGetData | — |
| PDXObjectGetDataLength | — |
| PDXObjectGetSubtype | — |

Cos Conversion:

[PDXObjectGetCosObj](#)

Validity testing: None

PDFEdit

Provides easy access to PDF page contents. With PDFEdit, you can treat a page's contents as a list of objects—rather than having to manipulate the content stream's PDF marking operators.

The PDFEdit API is meant to be used in conjunction with the Acrobat PModel and Cos APIs for manipulating PDF documents.

PDEClip

A list of *PDEElements* containing a list of *PDEPaths* and *PDETexts* that describe a clip state. *PDEClips* can be created and built up with *PDEClip* methods. Any *PDEElement* object can have *PDEClip* associated with it.

PDEClip objects can contain *PDEContainers* and *PDEGroups* to an arbitrary level of nesting. This allows *PDEContainers* to be used to mark clip objects.

PDEGroups inside *PDEClips* that contain at least one *PDEText* and no *PDEPaths* have a special meaning. All *PDEText* objects contained in such a *PDEGroup* are considered to be part of the same **BT/ET** block. This means that the union of these *PDETexts* makes up a single clipping path—as opposed to the *intersection* of the *PDETexts*.

Obtaining:

[PDEClipCreate](#)

[PDEElementGetClip](#)

Disposing:

[PDERelease](#)

Attributes:

| Get | Set |
|------------------------------------|------------------------------------|
| PDEClipGetElem | — |
| PDEClipGetNumElems | — |
| — | PDEClipAddElem |
| — | PDEClipRemoveElems |

Cos conversion: None

Validity testing: None

PDEColorSpace

A reference to a color space used on a page in a PDF file. The color space is part of the graphics state attributes of a *PDEElement*.

Obtaining:

[PDEColorSpaceCreate](#)[PDEColorSpaceCreateFromName](#)[PDEImageGetColorSpace](#)

Disposing:

[PDERelease](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|------------|
| PDEColorSpaceGetBase | — |
| PDEColorSpaceGetBaseNumComps | — |
| PDEColorSpaceGetCTable | — |
| PDEColorSpaceGetHiVal | — |
| PDEColorSpaceGetName | — |
| PDEColorSpaceGetNumComps | — |

Cos conversion:

[PDEColorSpaceCreate](#)[PDEColorSpaceGetCosObj](#)

Validity testing: None

Declarations:

[PDEColorSpec](#)[PDEColorValue](#)

PDEContainer

A group of *PDEElements* on a page in a PDF file. In the PDF file, containers are delimited by Marked Content **BMC/EMC** or **BDC/EMC** pairs. Every *PDEContainer* has a Marked Content tag associated with it. In addition to grouping a set of elements, a **BDC/EMC** pair specifies a property list to be associated with the grouping. Thus a *PDEContainer* corresponding to a **BDC/EMC** pair also has a property list dictionary associated with it.

Subclass of: [PDEElement](#)

Obtaining:

[PDEContainerCreate](#)

Disposing:

[PDERelease](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|--|
| PDEContainerGetContent | PDEContainerSetContent |
| PDEContainerGetDict | PDEContainerSetDict |
| PDEContainerGetMCTag | PDEContainerSetMCTag |

Cos conversion: None

Validity testing: None

PDEContent

Contains the modifiable contents of a *PDPage*. A *PDEContent* may be obtained from an existing page or from a Form *XObject* or from a Type 3 CharProc. You can create an empty *PDEContent*. A *PDEContent* contains *PDEElements*. In addition, a *PDEContent* may have attributes such as Form matrix and setcachedevice parameters.

Obtaining:

[PDEContentCreate](#)
[PDEContainerGetContent](#)
[PDEContentCreateFromCosObj](#)
[PDEFormGetContent](#)
[PDPageAcquirePDEContent](#)

Disposing:

[PDERelease](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|---|
| <u>PDEContentGetAttrs</u> | — |
| <u>PDEContentGetElem</u> | — |
| <u>PDEContentGetNumElems</u> | — |
| <u>PDEContentGetResources</u> | — |
| — | <u>PDEContentAddElem</u> |
| — | <u>PDEContentRemoveElem</u> |

Cos conversion:

[PDEContentCreateFromCosObj](#)
[PDEContentToCosObj](#)

Validity testing: None

Declarations:

[PDEContentAttrs](#)

[PDEContentFlags](#)

[PDEContentToCosObjFlags](#)

PDEDeviceNColors

A color space with a variable number of device-dependent components. Usually used to store multiple spot colors in a single color space.

Obtaining:

[PDEDeviceNColorsCreate](#)

Disposing: None

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|------------|
| <u>PDEDeviceNColorsGetColorValue</u> | — |

Cos conversion: None

Validity testing: None

PDEElement

The base class for elements of a page display list (*PDEContent*) and for clip objects. The general *PDEElement* methods allow you to get and set general element properties.

Subclasses:

[PDEContainer](#)

[PDEForm](#)

[PDEGroup](#)

[PDEImage](#)

[PDEPath](#)

[PDEPlace](#)

[PDEShading](#)

[PDEText](#)

[PDEUnknown](#)

[PDEXObject](#)

Obtaining:

[PDEClipGetElem](#)

[PDEContentGetElem](#)

[PDEElementCopy](#)

Disposing:

[PDERelease](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|--|
| <u>PDEElementGetBBox</u> | — |
| <u>PDEElementGetClip</u> | <u>PDEElementIsAtRect</u> |
| <u>PDEElementGetGState</u> | <u>PDEElementSetGState</u> |

| <i>Get</i> | <i>Set</i> |
|--|--|
| <u>PDEElementGetMatrix</u> | <u>PDEElementSetMatrix</u> |
| <u>PDEElementIsAtPoint</u> | — |
| <u>PDEElementIsAtRect</u> | — |

Cos conversion: None

Validity testing: None

Declarations:

[PDEElementCopyFlags](#)[PDEEnumElementsFlags](#)[PDEGraphicState](#)[PDEGraphicStateWasSetFlags](#)

PDEExtGState

A reference to an *ExtGState* resource used on a page in a PDF file. It specifies a *PDEElement*'s extended graphics state, which is part of its graphics state.

Obtaining:

[PDEExtGStateCreate](#)

Disposing:

[PDERelease](#)

Attributes: None

Cos conversion:

[PDEExtGStateGetCosObj](#)

Validity testing: None

PDEFont

A reference to a font used on a page in a PDF file. It may be equated with a font in the system. A *PDEFont* is not the same as a *PDFont*; a *PDFont* is associated with a particular document.

Obtaining:

[PDEFontCreate](#)
[PDEFontCreateFromCosObj](#)
[PDEFontCreateFromSysFont](#)
[PDEFontCreateWithParams](#)
[PDTextGetFont](#)

Disposing:

[PDERelease](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|------------|
| PDEFontCreateWithParams | — |
| PDEFontGetNumCodeBytes | — |
| PDEFontIsMultiByte | — |
| PDEFontSumWidths | — |
| PDFindSysFont | — |

Cos conversion:

[PDEFontCreateFromCosObj](#)
[PDEFontGetCosObj](#)

Validity testing: None

Declarations:

[PDEFontAttrs](#)

[PDEFontCreateFlags](#)

[PDEFontInfoRec](#)

PDEForm

A *PDEElement* that contains a Form *XObject*. A *PDEContent* may be obtained from a *PDEForm* to edit the form's display list.

Subclass of: [PDEElement](#)

Obtaining:

[PDEFormCreateFromCosObj](#)

Disposing:

[PDERelease](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|-----------------------------------|------------|
| PDEFormGetContent | — |

Cos conversion:

[PDEFormCreateFromCosObj](#)

[PDEFormGetCosObj](#)

Validity testing: None

PDEGroup

A *PDEElement* that specifies the beginning and ending of Marked Content in a PDF file. Any objects added to a *PDEGroup* object will be surrounded by the **BMC/EMC** marked content tags.

Obtaining:

[PDEGroupCreate](#)

Disposing: None

Attributes:

| <i>Get</i> | <i>Set</i> |
|------------------------------------|------------------------------------|
| PDEGroupGetContent | PDEGroupSetContent |

Cos conversion: None

Validity testing: None

PDEImage

A *PDEElement* that contains an Image *XObject* or inline image. You can associate data or a stream with an image.

Subclass of: [PDEElement](#)

Obtaining:

[PDEImageCreate](#)

[PDEImageCreateFromCosObj](#)

Disposing:

[PDERelease](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|------------------------------------|
| PDEImageDatalsEncoded | — |
| PDEImageGetAttrs | — |
| PDEImageGetColorSpace | — |
| PDEImageGetData | PDEImageSetData |
| PDEImageGetDataLen | — |
| PDEImageGetDataStm | PDEImageSetDataStm |
| PDEImageGetFilterArray | — |
| PDEImageIsCosObj | — |

Cos conversion:

[PDEImageCreateFromCosObj](#)

[PDEImageGetCosObj](#)

Validity testing: None

Declarations:

[PDEImageAttrFlags](#)

[PDEImageAttrs](#)

[PDEImageDataFlags](#)

PDEObject

The abstract superclass of *PDFEdit* classes. You can find the type of any object with the *PDEObjectGetType* method. You can then cast and apply that class's methods to the object. In addition, you can cast any of the *PDFEdit* objects to a *PDEObject* and use it anywhere a *PDEObject* is called for, such as in the *PDEObject* methods.

Obtaining: Various since all *PDFEdit* objects are *PDEObjects*.

Disposing:

[PDERelease](#)

Attributes:

| Get | Set |
|----------------------------------|------------------------------|
| PDEGetTag | PDEAddTag |
| — | PDERemoveTag |
| PDEObjectGetType | — |
| PDEObjectGetType | — |
| — | PDEAcquire |
| — | PDERelease |

Cos conversion: None

Validity testing: None

Declarations:

[PDEType](#)

PDEPath

A *PDEElement* that contains a path. Path objects can be stroked, filled, and/or serve as a clipping path.

Subclass of: [PDEElement](#)

Obtaining:

[PDEPathCreate](#)

Disposing:

[PDERelease](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|-----------------------------------|-----------------------------------|
| PDEPathGetData | PDEPathSetData |
| PDEPathGetPaintOp | PDEPathSetPaintOp |
| — | PDEPathAddSegment |

Cos conversion: None

Validity testing: None

Declarations:

[PDEPathElementType](#)

[PDEPathOpFlags](#)

PDEPattern

A reference to a Pattern resource used on a page in a PDF file.

Obtaining:

[PDEPatternCreate](#)

Disposing:

[PDERelease](#)

Attributes: None

Cos conversion:

[PDEPatternGetCosObj](#)

Validity testing: None

PDEPlace

A *PDEElement* that marks a place on a page in a PDF file. In a PDF file, a place is represented by the **MP** or **DP** Marked Content operators.

Marked content is useful for adding structure information to a PDF file. For instance, a drawing program may want to mark a point with information, such as the start of a path of a certain type. Marked content provides a way to retain this information in the PDF file. A **DP** operator functions the same as the **MP** operator and, in addition, allows a property list dictionary to be associated with a place.

Subclass of: [PDEElement](#)

Obtaining:

[PDEPlaceCreate](#)

Disposing:

[PDERelease](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|----------------------------------|----------------------------------|
| PDEPlaceGetDict | PDEPlaceSetDict |
| PDEPlaceGetMCTag | PDEPlaceSetMCTag |

Cos conversion: None

Validity testing: None

PDEShading

A *PDEElement* that represents smooth shading.

Obtaining:

[PDEShadingCreateFromCosObj](#)

Disposing: None

Attributes: None

Cos conversion:

[PDEShadingCreateFromCosObj](#)

[PDEShadingGetCosObj](#)

Validity testing: None

PDEText

A *PDEElement* representing text. It is a container for text as show strings or as individual characters. Each subelement may have different graphics state properties. However, the same clip applies to all subelements of a *PDEText*. Also, the *charpath* of a *PDEText* can be used to represent a clip.

Subclass of: [PDEElement](#)

Obtaining:

[PDETextCreate](#)

Disposing:

[PDERelease](#)

Attributes:

| Get | Set |
|---|---|
| PDETextGetAdvanceWidth | — |
| PDETextGetBBox | — |
| PDETextGetFont | PDETextRunSetFont |
| PDETextGetGState | PDETextRunSetGState |
| PDETextGetNumBytes | — |
| PDETextGetNumRuns | — |
| PDETextGetQuad | — |
| PDETextGetRunForChar | — |
| PDETextGetStrokeMatrix | PDETextRunSetStrokeMatrix |
| PDETextGetText | — |
| PDETextGetTextMatrix | PDETextRunSetTextMatrix |
| PDETextGetTextState | PDETextRunSetTextState |
| PDETextRunGetCharOffset | — |
| PDETextRunGetNumChars | — |

| Get | Set |
|----------------------------------|-------------------------------------|
| PDETextIsAtPoint | — |
| — | PDETextAdd |
| — | PDETextIsAtPoint |
| — | PDETextReplaceChars |
| — | PDETextSplitRunAt |

Cos conversion: None

Validity testing: None

Declarations:

- [PDEGraphicState](#)
- [PDEGraphicStateWasSetFlags](#)
- [PDETextFlags](#)
- [PDETextRenderMode](#)
- [PDETextState](#)
- [PDETextStateWasSetFlags](#)

PDEUnknown

A *PDEElement* representing an unknown element.

Subclass of: [PDEElement](#)

Obtaining: None

Disposing: None

Attributes:

| <i>Get</i> | <i>Set</i> |
|-------------------------------------|------------|
| PDEUnknownGetOpName | — |

Cos conversion: None

Validity testing: None

PDEXObject

A *PDEElement* representing an arbitrary *XObject*.

Subclass of: [PDEElement](#)

Obtaining:

[PDEXObjectCreate](#)

Disposing:

[PDERelease](#)

Attributes: None

Cos conversion:

[PDEXObjectGetCosObj](#)

Validity testing: None

PDSysFont

A reference to a font installed in the host system. *PDSysFont* methods allow you to list the fonts available in the host system and to find a font in the system that matches a *PDEFont*, if it is present.

Obtaining:

[PDFindSysFont](#)

[PDFindSysFontForPDEFont](#)

Disposing:

[PDERelease](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|------------|
| <u>PDSysFontAcquirePlatformData</u> | — |
| <u>PDSysFontGetCIDSystemInfo</u> | — |
| <u>PDSysFontGetEncoding</u> | — |
| <u>PDSysFontGetInfo</u> | — |
| <u>PDSysFontGetName</u> | — |
| <u>PDSysFontGetType0Widths</u> | — |

Cos conversion: None

Validity testing: None

Declarations:

[PDSysFontMatchFlags](#)

PDSEdit

The creation and manipulation of logical structure in PDF documents.

PDSAttrObj

Represents PDF logical structure attribute objects, which are dictionaries containing application-specific data that can be attached to *PDSElements*.

Obtaining:

[PDSAttrObjCreate](#)
[PDSAttrObjCreateFromStream](#)
[PDSCClassMapGetAttrObj](#)
[PDSElementGetAttrObj](#)

Disposing: None

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|--|
| <u>PDSAttrObjGetOwner</u> | — |
| <u>PDSCClassMapGetAttrObj</u> | <u>PDSCClassMapAddAttrObj</u> <u>PDSElementRemoveAttrObj</u> |
| <u>PDSElementGetAttrObj</u> | <u>PDSElementRemoveAttrObj</u> <u>PDSElementRemoveAllAttrObjs</u> |

Cos conversion: None

Validity testing: None

PDSCClassMap

Associates class identifiers, which are names, with objects of type *PDSAttrObj*. Structural elements maintain a list of names identifying classes to which they belong. The associated attributes are thus shared by all structural elements belonging to a given class. There is one class map per document, associated with the *PDSTreeRoot*.

Obtaining:

[PDSTreeRootCreateClassMap](#)

[PDSTreeRootGetClassMap](#)

Disposing: None

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|--|
| — | <u>PDSCClassMapAddAttrObj</u> |
| | <u>PDSCClassMapRemoveAttrObj</u> |
| <u>PDSCClassMapGetAttrObj</u> | — |
| <u>PDSCClassMapGetNumAttrObjs</u> | — |
| <u>PDSTreeRootGetClassMap</u> | <u>PDSTreeRootRemoveClassMap</u> |

Cos conversion: None

Validity testing: None

PDSElement

Represents PDF structural elements, which are nodes in a tree giving a PDF document's logical structure.

Obtaining:

[PDDocCreateStructTreeRoot](#)
[PDDocGetStructTreeRoot](#)
[PDSElementCreate](#)
[PDSElementGetParent](#)
[PDSMCGetParent](#)
[PDSOBJGetParent](#)

[PDSTreeRootGetElementFromID](#)

Disposing: None

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|--|
| — | <u>PDSElementAddAttrObj</u> |
| | <u>PDSElementRemoveAttrObj</u> |
| | <u>PDSElementRemoveAllAttrObjs</u> |
| — | <u>PDSElementAddClass</u> |
| | <u>PDSElementRemoveClass</u> |
| | <u>PDSElementRemoveAllClasses</u> |
| <u>PDSElementGetAlt</u> | <u>PDSElementSetAlt</u> |
| <u>PDSElementGetAttrObj</u> | — |
| <u>PDSElementGetClass</u> | — |
| <u>PDSElementGetFirstPage</u> | — |
| <u>PDSElementGetID</u> | <u>PDSElementClearID</u> |
| | <u>PDSElementSetID</u> |
| <u>PDSElementGetKid</u> | <u>PDSElementReplaceKid</u> |
| <u>PDSElementGetNumAttrObjs</u> | — |
| <u>PDSElementGetNumClasses</u> | — |
| <u>PDSElementGetNumKids</u> | — |

| Get | Set |
|---------------------------------------|---|
| PDSElementGetParent | — |
| PDSElementGetRevision | PDSElementIncrementRevision |
| PDSElementGetTitle | PDSElementSetTitle |
| PDSElementGetType | PDSElementSetType |
| — | PDSElementInsertKid |
| — | PDSElementRemoveKid |
| — | PDSElementInsertMCAsKid |
| | PDSElementRemoveKidMC |
| | PDSElementReplaceKidMC |

Cos conversion: None

Validity testing: None

PDSMC

Represents marked content—portions of the graphic content of a PDF document that may be included in the document's logical structure hierarchy. This type is identical with the PDFEdit layer type *PDEContainer*.

Obtaining: None

Attributes:

| <i>Get</i> | <i>Set</i> |
|--------------------------------------|------------|
| PDSElementReplaceKid | — |
| PDSMCGetParent | — |

Cos conversion: None

Validity testing: None

PDSRoleMap

Represents mappings of structural element types present in a PDF document to standard element types having similar uses. There is one *PDSClassMap* per document, associated with the *PDSTreeRoot*.

Obtaining:

[PDSRoleMapCopy](#)

[PDSTreeRootCreateRoleMap](#)

[PDSTreeRootGetRoleMap](#)

Disposing: None

Attributes:

| <i>Get</i> | <i>Set</i> |
|---|---|
| <u>PDSRoleMapGetDirectMap</u> | — |
| — | <u>PDSRoleMapMap</u> |
| <u>PDSTreeRootGetRoleMap</u> | <u>PDSTreeRootRemoveRoleMap</u> |

Cos conversion: None

Validity testing: None

PDSTreeRoot

Forms a central repository for information related to a PDF document's logical structure. There is at most one *PDSTreeRoot* in each document.

Obtaining:

[PDDocCreateStructTreeRoot](#)

Disposing:

[PDDocRemoveStructTreeRoot](#)

Attributes:

| <i>Get</i> | <i>Set</i> |
|--|--|
| <u>PDDocGetStructTreeRoot</u> | — |
| <u>PDSTreeRootGetClassMap</u> | <u>PDSTreeRootRemoveClassMap</u> |
| <u>PDSTreeRootGetElementFromID</u> | — |
| <u>PDSTreeRootGetKid</u> | <u>PDSTreeRootInsertKid</u> <u>PDSTreeRootRemoveKid</u> |
| <u>PDSTreeRootGetNumKids</u> | — |
| <u>PDSTreeRootGetRoleMap</u> | <u>PDSTreeRootRemoveRoleMap</u> |
| — | <u>PDSTreeRootReplaceKid</u> |

Cos conversion: None

Validity testing: None

API Changes

Version 2.1

Methods

The following methods were added in version 2.1.

Available only if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020001 or higher.

[AVDocGetClientName](#)

[AVDocGetPageText](#)

[AVDocSetClientName](#)

Available only if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020001 or higher.

[PDDocClearFlags](#)

[PDDrawCosObjToWindow](#)

[PDPageNotifyContentsDidChangeEx](#)

Errors

The following errors were added in version 2.1.

[xmErrPluginLoadFailed](#)

[xmErrNotPrivileged](#)

[xmErr68KOnly](#)

[xmErrPPCOnly](#)

[avErrNoText](#)

Notifications

The following notifications were added in version 2.1.

[PDPageContentsDidChangeEx](#)

Version 3.0

Objects

The following object was added in version 3.0.

[PDTrans](#)

Methods

The following methods were added in version 3.0.

Available only if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

[ASFileFromMDFile](#)
[ASFileGetFileSysByName](#)
[ASFileGetMDFile](#)
[ASFilePushData](#)
[ASFileRegisterFileSys](#)
[ASFileSetMode](#)
[ASFileSysAcquireFileSysPath](#)
[ASFileSysCreatePathName](#)
[ASFileUnregisterFileSys](#)

Available only if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

[AVAppEnumTransHandlers](#)
[AVAppGetTransHandlerByType](#)
[AVAppHandlePlatformEvent](#)
[AVAppRegisterTransHandler](#)
[AVAuthOpen](#)
[AVDocDoActionPropsDialog](#)
[AVDocIsExternal](#)
[AVDocDoSaveAs](#)
[AVDocGetViewDef](#)
[AVDocIsReadOnly](#)
[AVDocPrintPagesWithParams](#)
[AVDocSendAuxData](#)
[AVDocSetDead](#)

[AVDocSetViewDef](#)
[AVHasAuxDataHandler](#)
[AVPageViewDrawCosObj](#)
[AVPageViewGetFirstVisiblePageNum](#)
[AVPageViewGetLastVisiblePageNum](#)
[AVPageViewGetLayoutMode](#)
[AVPageViewGetNextView](#)
[AVPageViewGetSelectedAnnotPageNum](#)
[AVPageViewHighlightText](#)
[AVPageViewInsetRect](#)
[AVPageViewInvalidateText](#)
[AVPageViewPageNumIsVisible](#)
[AVPageViewPointInText](#)
[AVPageViewSetLayoutMode](#)
[AVPageViewTrackText](#)
[AVPageViewSetPageNum](#)
[AVPageViewUseThisDestination](#)
[AVRegisterAuxDataHandler](#)
[AVToolButtonSetExternal](#)
[AVToolButtonSetHelpText](#)
[AVUnregisterAuxDataHandler](#)
[AVWindowGetCursorAtPoint](#)
[AVAppWindowHandlePlatformEvent](#)

Available only if *PI_MACINTOSH_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

[AVAppEnumSystemFonts](#)

Available only if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

[CosDocClose](#)
[CosDocCreate](#)
[CosDocOpenWithParams](#)
[CosDocSaveToFile](#)
[CosDocSaveWithParams](#)

Available only if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

[PDDocGetCryptHandlerClientData](#)
[PDDocGetFullScreen](#)
[PDDocOpenEx](#)
[PDDocOpenFromASFile](#)
[PDDocOpenFromASFileEx](#)
[PDDocReadAhead](#)
[PDDocSaveWithParams](#)
[PDDocSetFullScreen](#)
[PDFileSpecGetDoc](#)
[PDFileSpecGetFileSysName](#)
[PDNameTreeLookup](#)
[PDPageGetDuration](#)
[PDPageGetTransition](#)
[PDPageHasTransition](#)
[PDPageSetDuration](#)
[PDPageSetTransition](#)
[PDPageStmGetInlinelImage](#)
[PDPageStmGetToken](#)
[PDRegisterCryptHandlerEx](#)
[PDRegisterFileSpecHandlerByName](#)
[PDTransEqual](#)
[PDTransFromCosObj](#)
[PDTransGetCosObj](#)
[PDTransGetDuration](#)
[PDTransGetSubtype](#)
[PDTransNew](#)
[PDTransNewFromCosDoc](#)
[PDTransNull](#)
[PDTransIsValid](#)
[PDViewDestResolve](#)

Available only if *PI_WIN_VERSION* (in *PIRequir.h*) is set to 0x00020000 or higher.

[WinAppEnableIdleTimer](#)
[WinAppGetModalParent](#)
[WinAppGetPalette](#)
[WinAppRegisterModelessDialog](#)
[WinAppUnRegisterModelessDialog](#)

The following methods were added to the Acrobat Toolkit.

[ASGetErrorString](#)
[ASfree](#)
[ASmalloc](#)
[ASrealloc](#)
[CosDictPut](#)
[CosDictRemove](#)
[CosNewString](#)
[PDDocAuthorize](#)
[PDDocCreate](#)
[PDDocImportNotes](#)
[PDDocSaveWithParams](#)
[PDFontAcquireXlateTable](#)
[PDFontXlateTableRelease](#)
[PDFontXlateString](#)
[PDFontXlateWidths](#)
[PDPageGetDoc](#)
[PDRegisterCryptHandler](#)
[PDXlateToPDFDocEnc](#)

The following methods are now replaceable.

[AVDocDoSaveAs](#)
[AVPageViewGetNextView](#)
[PDDocSave](#)
[PDDocSaveWithParams](#)

Callbacks

The following callbacks were added in version 3.0.

[ASFileCompletionProc](#)
[ASFileSysAcquireFileSysPathProc](#)
[ASFileSysAsyncAbortProc](#)
[ASFileSysAsyncReadProc](#)
[ASFileSysAsyncWriteProc](#)
[ASFileSysClearOutstandingMReadsProc](#)
[ASFileSysCreatePathNameProc](#)
[ASFileSysGetFileFlags](#)
[ASFileSysGetStatusProc](#)
[ASFileSysMReadRequestProc](#)
[ASFileSysYieldProc](#)
[ASIODoneProc](#)
[ASMemFreeProc](#)
[AVAnnotHandlerCursorEnterProc](#)
[AVAnnotHandlerCursorExitProc](#)
[AVAuxDataPerformProc](#)
[AVTransHandlerCompleteTransDictProc](#)
[AVTransHandlerDoPropertiesProc](#)
[AVTransHandlerEnumProc](#)
[AVTransHandlerExecuteProc](#)
[AVTransHandlerGetButtonTextProc](#)
[AVTransHandlerGetInstructionsProc](#)
[AVTransHandlerGetStringOneTextProc](#)
[AVTransHandlerGetStringTwoTextProc](#)
[AVTransHandlerGetTypeProc](#)
[AVTransHandlerGetItemUINameProc](#)
[AVTransHandlerGetUINameProc](#)
[AVTransHandlerInitTransDictProc](#)
[PDAuthProcEx](#)
[PDCryptFreeAuthDataProc](#)
[PDCryptFreeSecurityDataProc](#)
[PDLaunchActionProc](#)
[PDPageStmImageDataProc](#)
[PDPageStmStringOverflowProc](#)

Data

The following data structures were added in version 3.0.

- [ASFile Flags](#)
- [ASFile Open Modes](#) (*ASFILE_SERIAL* and *ASFILE_LOCAL*)
- [ASFileStatus Flags](#)
- [ASFileSysRec](#) (significantly expanded)
- [ASIORequest](#)
- [ASPlatformPrinterSpec](#)
- [AVAnnotHandler](#) (new callbacks)
- [AVAuxDataHandler](#)
- [AVDocOpenParams](#) (significantly expanded)
- [AVDocPrintParams](#)
- [AVDocViewDef](#)
- [AVPrefsType](#) (significantly expanded)
- [AVSystemFont](#)
- [AVSystemFont Flags](#)
- [AVTransHandler](#)
- [AVTransitionPort](#)
- [CosDocCreate Flags](#)
- [CosDocSave Flags](#)
- [CosDocSaveParams](#)
- [EmitFontOptions](#)
- [Emit Flags](#)
- [Page Specification](#)
- [PDFCryptHandler](#) (new callbacks)
- [PDDocReadAhead Flags](#)
- [PDLayoutMode](#)
- [PDPageStmToken](#)
- [Predefined Cursors](#) (cursors added)
- [Tool Button Flags](#)
- [Transition Duration](#)

Errors

The following error system was added in version 3.0.

[ErrSysXtn](#)

The following errors were added in version 3.0.

[cfMacGenPSErr](#)
[cfMaciPrSavPFil](#)
[cfMacServerLostConnection](#)
[cosErrBadIndex](#)
[cosErrCancelSave](#)
[cosErrOldLinFormat](#)
[cosErrTempTooShort](#)
[cosSynErrBadLinearized](#)
[fileErrBytesNotReady](#)
[fileErrUserRequestedStop](#)
[fileErrIOTimeout](#)
[fsErrBadParameter](#)
[fsErrDownloadAborted](#)
[fsErrDownloadFailed](#)
[pageErrBadColorSpace](#)
[pageErrBadEGS](#)
[pageErrBadPattern](#)
[pageErrEGStateNotFound](#)
[pageErrMissingKey](#)
[pageErrMissingResource](#)
[pageErrPatternNotFound](#)
[pageErrPatternTypeNotAvailable](#)
[pdErrATMMemory](#)
[pdErrBadCMap](#)
[pdErrCancelSave](#)
[pdErrCannotReopenDoc](#)
[pdErrOldATMVersion](#)
[pdErrOptMemory](#)
[pdErrTextStringTooShort](#)
[pdErrZeroPageFile](#)

Notifications

The following notifications were added in version 3.0.

Available only if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020002 or higher.

[AVDocDidAddToSelection](#)

[AVDocWantsToDie](#)

[AVDocWillOpenFromPDDoc](#)

[PDDocWillPrintDoc](#)

[PDDocWillSaveEx](#)

Miscellaneous

Values of some types in methods, callbacks, and data have been changed for improved cross-platform portability.

| <i>Previous name</i> | <i>New name</i> |
|----------------------|-----------------|
| <i>boolean</i> | <i>ASBool</i> |
| <i>Int8</i> | <i>ASInt8</i> |
| <i>Int16</i> | <i>ASInt16</i> |
| <i>Int32</i> | <i>ASInt32</i> |
| <i>Uns8</i> | <i>ASUns8</i> |
| <i>Uns16</i> | <i>ASUns16</i> |
| <i>Uns32</i> | <i>ASUns32</i> |

Version 3.0J

Methods

The following methods were added in version 3.0J.

Available only if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

[PDGetHostEncoding](#)
[PDHostMBLen](#)
[PDXlateToHostEx](#)
[PDXlateToPDFDocEncEx](#)
[PDDocCreateWordFinderUCS](#)
[PDFontGetCIDSystemInfo](#)
[PDFontGetCIDSystemSupplement](#)
[PDFontGetDescendant](#)
[PDFontGetEncodingName](#)
[PDFontXlateToHost](#)
[PDFontXlateToUCS](#)

The following methods have new behavior in version 3.0J.

[PDWordFilterString](#)
[PDWordGetString](#)
[PDWordSplitString](#)

Version 3.01

Methods

The following methods were added in version 3.01.

Available only if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00020003 or higher.

[AVDestInfoDestroy](#)
[AVDocCopyAction](#)
[AVDocCopyActionCommon](#)
[AVDocCopyAdditionalActions](#)
[AVDocCopyAnnot](#)
[AVDocCopyAnnotCommon](#)
[AVDocDoCopyAs](#)
[AVPageViewDrawCosObjEx](#)
[AVPageViewToDestInfo](#)
[AVPageViewUseDestInfo](#)

Callbacks

The following callbacks were added in version 3.01.

[AVActionCopyProc](#)
[AVAnnotHandlerCopyProc](#)
[CrossDocLinkWithDestProc](#)

Data

The following data structures were added or changed in version 3.01.

[AVAnnotHandler](#) (new flag)
[AVDestInfo](#)
[EmitFontOptions](#) (obsoleted and new flags)
[ExternalDocServerCreationData](#) (new callback and data)

Errors

The following errors were added in version 3.01.

[avErrBadActionCopy](#)
[avErrBadAnnotationCopy](#)

Adobe PDF Library, Version 1.0

All API elements listed here are available only if the Adobe PDF Library Version is 1.0 or later. This version adds the PDFEdit methods.

Objects

The following objects were added for PDFEdit.

[PDEClip](#)
[PDEColorSpace](#)
[PDEContainer](#)
[PDEContent](#)
[PDEElement](#)
[PDEExtGState](#)
[PDEFont](#)
[PDEForm](#)
[PDEImage](#)
[PDEObject](#)
[PDEPath](#)
[PDEPattern](#)
[PDEPlace](#)
[PDEText](#)
[PDEXObject](#)
[PDSysFont](#)

Methods

The following methods were added for the Adobe PDF Library.

[PDDocPrintPages](#)
[PDFLInit](#)
[PDFLPrintDoc](#)
[PDFLTerm](#)
[AVExtensionMgrRegisterNotification](#)
[AVExtensionMgrUnregisterNotification](#)
[PDSetHostEncoding](#)

The following methods were added for PDFEdit.

[PDELogDump](#)
[PDEObjectDump](#)
[PDEAttrEnumTable](#)
[PDEDefaultGState](#)
[PDEEnumElements](#)
[PDEMergeResourcesDict](#)
[PDEClipAddElem](#)
[PDEClipCreate](#)
[PDEClipGetElem](#)
[PDEClipGetNumElems](#)
[PDEClipRemoveElems](#)
[PDEColorSpaceCreate](#)
[PDEColorSpaceCreateFromName](#)
[PDEColorSpaceGetBase](#)
[PDEColorSpaceGetBaseNumComps](#)
[PDEColorSpaceGetCosObj](#)
[PDEColorSpaceGetCTable](#)
[PDEColorSpaceGetHiVal](#)
[PDEColorSpaceGetName](#)
[PDEColorSpaceGetNumComps](#)
[PDEContainerCreate](#)
[PDEContainerGetContent](#)
[PDEContainerGetDict](#)
[PDEContainerGetMCTag](#)
[PDEContainerSetContent](#)
[PDEContainerSetDict](#)
[PDEContainerSetMCTag](#)
[PDEContentAddElem](#)

[PDEContentCreate](#)
[PDEContentCreateFromCosObj](#)
[PDEContentGetAttrs](#)
[PDEContentGetElem](#)
[PDEContentGetNumElems](#)
[PDEContentGetResources](#)
[PDEContentRemoveElem](#)
[PDEContentToCosObj](#)
[PDEElementCopy](#)
[PDEElementGetBBox](#)
[PDEElementGetClip](#)
[PDEElementGetGState](#)
[PDEElementGetMatrix](#)
[PDEElementSetGState](#)
[PDEElementSetMatrix](#)
[PDEExtGStateCreate](#)
[PDEExtGStateGetCosObj](#)
[PDEFontCreate](#)
[PDEFontCreateFromCosObj](#)
[PDEFontCreateFromSysFont](#)
[PDEFontCreateWithParams](#)
[PDEFontGetCosObj](#)
[PDEFontGetNumCodeBytes](#)
[PDEFontSubsetNow](#)
[PDFindSysFontForPDEFont](#)
[PDEFormCreateFromCosObj](#)
[PDEFormGetContent](#)
[PDEFormGetCosObj](#)
[PDEImageCreate](#)
[PDEImageCreateFromCosObj](#)
[PDEImageDataIsEncoded](#)
[PDEImageGetAttrs](#)
[PDEImageGetColorSpace](#)
[PDEImageGetCosObj](#)
[PDEImageGetData](#)
[PDEImageGetDataLen](#)
[PDEImageGetDataStm](#)
[PDEImageGetFilterArray](#)
[PDEImageIsCosObj](#)
[PDEImageSetData](#)
[PDEImageSetDataStm](#)
[PDEAcquire](#)
[PDEAddTag](#)

[PDEGetTag](#)
[PDEObjectGetType](#)
[PDERelease](#)
[PDERemoveTag](#)
[PDEPathAddSegment](#)
[PDEPathCreate](#)
[PDEPathGetData](#)
[PDEPathGetPaintOp](#)
[PDEPathSetData](#)
[PDEPathSetPaintOp](#)
[PDEPatternCreate](#)
[PDEPatternGetCosObj](#)
[PDEPlaceCreate](#)
[PDEPlaceGetDict](#)
[PDEPlaceGetMCTag](#)
[PDEPlaceSetDict](#)
[PDEPlaceSetMCTag](#)
[PDETextAdd](#)
[PDETextCreate](#)
[PDETextGetAdvanceWidth](#)
[PDETextGetBBox](#)
[PDETextGetFont](#)
[PDETextGetGState](#)
[PDETextGetNumBytes](#)
[PDETextGetNumRuns](#)
[PDETextGetQuad](#)
[PDETextGetRunForChar](#)
[PDETextGetStrokeMatrix](#)
[PDETextGetText](#)
[PDETextGetTextMatrix](#)
[PDETextGetTextState](#)
[PDETextIsAtPoint](#)
[PDETextReplaceChars](#)
[PDETextRunGetCharOffset](#)
[PDETextRunGetNumChars](#)
[PDETextRunSetFont](#)
[PDETextRunSetGState](#)
[PDETextRunSetStrokeMatrix](#)
[PDETextRunSetTextMatrix](#)
[PDETextRunSetTextState](#)
[PDETextSplitRunAt](#)
[PDEXObjectCreate](#)
[PDEXObjectGetCosObj](#)

[PDEnumSysFonts](#)
[PDFindSysFont](#)
[PDPageAcquirePDEContent](#)
[PDPageGetPDEContentFilters](#)
[PDPageGetPDEContentFlags](#)
[PDPagePDEContentWasChanged](#)
[PDPageRegisterForPDEContentChanged](#)
[PDPageRegisterForPDEContentNotCached](#)
[PDPageReleasePDEContent](#)
[PDPageSetPDEContent](#)
[PDPageSetPDEContentFilters](#)
[PDPageSetPDEContentFlags](#)
[PDPageUnRegisterForPDEContentChanged](#)
[PDPageUnRegisterForPDEContentNotCached](#)
[PDSysFontAcquirePlatformData](#)
[PDSysFontGetEncoding](#)
[PDSysFontGetInfo](#)
[PDSysFontGetName](#)
[PDSysFontGetType0Widths](#)

Callbacks

The following callbacks were added for PDFEdit.

[PDEAttrEnumProc](#)
[PDEElementEnumProc](#)
[PDEObjectDumpProc](#)
[PDSysFontEnumProc](#)

Data

The following data structures were added, mostly for PDFEdit.

[PDFLData](#) (for Adobe PDF Library)

[PDEColorSpec](#)

[PDEColorValue](#)

[PDEContentAttrs](#)

[PDEContentFlags](#)

[PDEContentToCosObjFlags](#)

[PDEDash](#)

[PDEElementCopyFlags](#)

[PDEEnumElementsFlags](#)

[PDEFilterArray](#)

[PDEFilterSpec](#)

[PDEFontAttrs](#)

[PDEFontCreateFlags](#)

[PDEFontInfoRec](#)

[PDEGraphicState](#)

[PDEGraphicStateWasSetFlags](#)

[PDEImageAttrFlags](#)

[PDEImageAttrs](#)

[PDEImageDataFlags](#)

[PDEPathElementType](#)

[PDEPathOpFlags](#)

[PDETextFlags](#)

[PDETextRenderMode](#)

[PDETextState](#)

[PDETextStateWasSetFlags](#)

[PDEType](#)

[PDSysFontMatchFlags](#)

Notifications

The following notifications were added for PDFEdit.

[PagePDEContentDidChange](#)

[PagePDEContentNotCached](#)

Version 4.0

Objects

The following objects were added in version 4.0.

[ASExtension](#)
[PDEDeviceNColors](#)
[PDEGroup](#)
[PDEShading](#)
[PDEUnknown](#)
[PDNameTree](#)
[PDNumTree](#)
[PDPageLabel](#)
[PDSAttrObj](#)
[PDSCClassMap](#)
[PDSElement](#)
[PDSMC](#)
[PDSRoleMap](#)
[PDSTreeRoot](#)

Methods

The following methods were obsoleted in version 4.0.

[PDPageEnumContents](#)
[PDPageEnumResources](#)

The following methods were added in version 4.0.

Available only if *PI_ACROSUPPORT_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

[ASEnumExtensions](#)
[ASExtensionGetFileName](#)
[ASExtensionGetRegisteredName](#)
[ASFileStmWrOpen](#)
[ASProcStmWrOpen](#)
[HFTIsValid](#)
[WinAppGetPrinterHDC](#)

Available only if *PI_ACROVIEW_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

[AVAnnotHandlerDeleteInfo](#)
[AVAnnotHandlerGetInfo](#)
[AVAppHandlePlatformEvent](#)
[AVAppOpenHelpFile](#)
[AVDocAlert](#)
[AVDocAlertConfirm](#)
[AVDocAlertNote](#)
[AVDocAlertYesNo](#)
[AVDocDoCopyAs](#)
[AVDocDoPrint](#)
[AVDocDoSaveAsWithParams](#)
[AVDocsReadOnly](#)
[AVDocSelectionEnumPageRanges](#)
[AVDocSetReadOnly](#)
[AVMenuIsHiddenOnMenubar](#)
[AVMenubarAddHiddenMenu](#)
[AVPageViewAppearanceGetAVMatrix](#)
[AVPageViewDeviceRectToPageRZ](#)
[AVPageViewDoPopupMenu](#)
[AVPageViewDrawAnnotSequence](#)
[AVPageViewGetGrayRect](#)
[AVPageViewGetVisibleAnnotPage](#)
[AVPageViewInvertQuad](#)
[AVPageViewShowControl](#)
[AVPageViewTransformRectRZ](#)
[AVSysAllocTimeStringFromTimeRec](#)
[AVToolBarNewFlyout](#)
[AVToolButtonGetFlyout](#)
[AVToolButtonGetIcon](#)
[AVToolButtonGetMenu](#)
[AVToolButtonSetFlyout](#)
[AVToolButtonSetIcon](#)
[AVToolButtonSetMenu](#)
[AVWindowHandlePlatformEvent](#)

Available only if *PI_COS_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

[CosArrayRemoveNth](#)
[CosDocEnumEOFs](#)
[CosDocEnumIndirect](#)
[CosDocGetObjByID](#)
[CosDocSaveWithParams](#)
[CosObjCopy](#)
[CosObjGetGeneration](#)
[CosObjGetID](#)
[CosObjHash](#)
[CosStringGetHexFlag](#)
[CosStringSetHexFlag](#)

Available only if *PI_PDMODEL_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

[PDDocCreateNameTree](#)
[PDDocCreateStructTreeRoot](#)
[PDDocEnumResources](#)
[PDDocExportNotes](#)
[PDDocFindPageNumForLabel](#)
[PDDocFromCosDoc](#)
[PDDocGetLabelForPageNum](#)
[PDDocGetNameTree](#)
[PDDocGetPageLabel](#)
[PDDocGetStructTreeRoot](#)
[PDDocImportCosDocNotes](#)
[PDDocImportNotes](#)
[PDDocOpenWithParams](#)
[PDDocReadAheadPages](#)
[PDDocRemoveNameTree](#)
[PDDocRemovePageLabel](#)
[PDDocRemoveStructTreeRoot](#)
[PDDocSetPageLabel](#)
[PDFontFromCosObj](#)
[PDGetAnnotHandlerByName](#)
[PDImageSelectAlternate](#)
[PDImageSelAdjustMatrix](#)
[PDImageSelGetDeviceAttr](#)
[PDImageSelGetGeoAttr](#)
[PDNameTreeEnum](#)

[PDNameTreeEqual](#)
[PDNameTreeFromCosObj](#)
[PDNameTreeGet](#)
[PDNameTreeGetCosObj](#)
[PDNameTreesValid](#)
[PDNameTreeNew](#)
[PDNameTreePut](#)
[PDNameTreeRemove](#)
[PDPageGetAnnotSequence](#)
[PDPageLabelEqual](#)
[PDPageLabelFromCosObj](#)
[PDPageLabelGetCosObj](#)
[PDPageLabelGetPrefix](#)
[PDPageLabelGetStart](#)
[PDPageLabelGetStyle](#)
[PDPageLabellsValid](#)
[PDPageLabelNew](#)
[PDRegisterAnnotHandler](#)

The following methods were added for the Adobe PDF Library.

[ASAtomGetCount](#)
[ASPurgeMemory](#)
[PDFLPrintDoc](#)

The following methods were added for Placed PDF.

[AGMCleanup](#)
[AGMClip](#)
[AGMClosePath](#)
[AGMConcat](#)
[AGMDeletePort](#)
[AGMDeleteRasterDev](#)
[AGMFill](#)
[AGMInit](#)
[AGMLineTo](#)
[AGMMoveTo](#)
[AGMNewBitmapPort](#)
[AGMNewPath](#)
[AGMNewRasterDev](#)
[AGMNewRasterPort](#)
[AGMNewWindowPort](#)
[AGMSetAntiAliasPolicy](#)
[AGMSetRGBColor](#)
[CCSetSystemCalibration](#)
[PDDocPrintPages](#)
[PDFontDownloadContextCreate](#)
[PDFontDownloadContextDestroy](#)
[PDFontStreamPS](#)
[PDFontWasExtracted](#)
[PDFontWasFauxed](#)
[PDFreeMemory](#)
[PDPageDrawContents](#)
[PDPageDrawContentsPlaced](#)
[PDPageDrawContentsPlacedToWindow](#)
[PDPageEmitPSOrient](#)

The following methods were added for PDFEdit.

Available only if *PI_PDFEDIT_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

[PDEClipFlattenedEnumElems](#)
[PDEContentGetDefaultColorSpace](#)
[PDEDeviceNColorsGetColorValue](#)
[PDEElementIsAtPoint](#)
[PDEElementIsAtRect](#)
[PDEFontGetNumCodeBytes](#)
[PDEFontGetOneByteEncoding](#)
[PDEFontIsMultiByte](#)
[PDEFontSumWidths](#)
[PDEGroupGetContent](#)
[PDETextGetNumBytes](#)
[PDETextIsAtPoint](#)
[PDETextIsAtRect](#)

Available only if *PI_PDFEDIT_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

[PDEClipAddElem](#)
[PDEClipCopy](#)
[PDEClipCreate](#)
[PDEColorSpaceCreate](#)
[PDEDeviceNColorsCreate](#)
[PDEFontCreateFromSysFontEx](#)
[PDEFontCreateWithParams](#)
[PDEGroupCreate](#)
[PDEGroupSetContent](#)
[PDEImageGetDecodeArray](#)
[PDEImageSetDecodeArray](#)
[PDEPurgeCache](#)
[PDEShadingCreateFromCosObj](#)
[PDEUnknownGetOpName](#)

Available only if *PI_PDSYSFONT_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

[PDEmbedSysFontForPDEFont](#)
[PDFindSysFontEx](#)
[PDSysFontAcquirePlatformData](#)
[PDSysFontGetCIDSystemInfo](#)
[PDSysFontGetType0Widths](#)
[PDSysFontGetWidthsEx](#)
[PDSysFontReleasePlatformData](#)

The following methods were added for PDS (Structure level).

Available only if *PI_PDS_READ_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

[PDSAttrObjGetOwner](#)
[PDSCClassMapGetAttrObj](#)
[PDSCClassMapGetNumAttrObjs](#)
[PDSElementGetAlt](#)
[PDSElementGetAttrObj](#)
[PDSElementGetClass](#)
[PDSElementGetFirstPage](#)
[PDSElementGetID](#)
[PDSElementGetKid](#)
[PDSElementGetNumAttrObjs](#)
[PDSElementGetNumClasses](#)
[PDSElementGetNumKids](#)
[PDSElementGetParent](#)
[PDSElementGetRevision](#)
[PDSElementGetStructTreeRoot](#)
[PDSElementGetTitle](#)
[PDSElementGetType](#)
[PDSMCGetParent](#)
[PDSOBJGetParent](#)
[PDSRoleMapDoesMap](#)
[PDSRoleMapGetDirectMap](#)
[PDSTreeRootGetClassMap](#)
[PDSTreeRootGetElementFromID](#)
[PDSTreeRootGetKid](#)
[PDSTreeRootGetNumKids](#)
[PDSTreeRootGetRoleMap](#)

Available only if *PI_PDS_WRITE_VERSION* (in *PIRequir.h*) is set to 0x00040000 or higher.

[PDSAttrObjCreate](#)
[PDSAttrObjCreateFromStream](#)
[PDSClassMapAddAttrObj](#)
[PDSClassMapRemoveAttrObj](#)
[PDSClassMapRemoveClass](#)
[PDSElementAddAttrObj](#)
[PDSElementAddClass](#)
[PDSElementClearID](#)
[PDSElementCreate](#)
[PDSElementIncrementRevision](#)
[PDSElementInsertKid](#)
[PDSElementInsertMCAsKid](#)
[PDSElementInsertOBJAsKid](#)
[PDSElementRemoveAttrObj](#)
[PDSElementRemoveAllAttrObjs](#)
[PDSElementRemoveAllClasses](#)
[PDSElementRemoveClass](#)
[PDSElementRemoveKid](#)
[PDSElementRemoveKidMC](#)
[PDSElementReplaceKid](#)
[PDSElementReplaceKidMC](#)
[PDSElementReplaceKidOBJ](#)
[PDSElementSetAlt](#)
[PDSElementSetID](#)
[PDSElementSetTitle](#)
[PDSElementSetType](#)
[PDSRoleMapCopy](#)
[PDSRoleMapMap](#)
[PDSRoleMapUnMapDst](#)
[PDSRoleMapUnMapSrc](#)
[PDSTreeRootCreateClassMap](#)
[PDSTreeRootCreateRoleMap](#)
[PDSTreeRootInsertKid](#)
[PDSTreeRootRemoveClassMap](#)
[PDSTreeRootRemoveKid](#)
[PDSTreeRootRemoveRoleMap](#)
[PDSTreeRootReplaceKid](#)

Callbacks

The following callbacks were added in version 4.0.

[ASExtensionEnumProc](#)
[ASProcStmDestroyProc](#)
[ASStmProc](#)
[AVAnnotHandlerDeleteInfoProc](#)
[AVAnnotHandlerGetInfoProc](#)
[AVDocSelectionEnumPageRangesProc](#)
[AVDocSelectionGetSelectionTypeProc](#)
[AVSelectionPageRangeEnumProc](#)
[AVWindowDestroyPlatformThingProc](#)
[CosDocEnumEOFsProc](#)
[CosObjOffsetProc](#)
[CosObjSetCallbackFlagProc](#)
[DoClickProcType](#)
[DoLeaveProcType](#)
[GetSelectionServerProcType](#)
[PDAnnotHandlerDeleteAnnotInfoProc](#)
[PDAnnotHandlerGetAnnotInfoFlagsProc](#)
[PDAnnotHandlerGetAnnotInfoProc](#)
[PDAnnotHandlerGetTypeProc](#)
[PDAnnotWillPrintProc](#)
[PDDocPreSaveProc](#)
[PDDocWillExportAnnotCallback](#)
[PDDocWillExportAnnotProc](#)
[PDDocWillImportAnnotCallback](#)
[PDDocWillImportAnnotProc](#)
[PDEClipEnumProc](#)
[PIExportHFTsProcType](#)
[PIHandshake](#)
[PIImportReplaceAndRegisterProcType](#)
[PIInitProcType](#)
[PIUnloadProcType](#)

The following callbacks were added for the Adobe PDF Library.

[ASMemAllocProc](#)
[ASMemAvailProc](#)
[ASMemFreeProc](#)
[ASMemReallocProc](#)
[PDFLPrintCancelProc](#)
[TKResourceAcquireProc](#)
[TKResourceReleaseProc](#)

The following callbacks were added for Placed PDF.

[AGMMemAllocator](#)
[AGMMemDeleter](#)
[AGMPortDestructProcPtr](#)
[DoCancel](#)
[DocBegin](#)
[DocEnd](#)
[DocSetup](#)
[EmitPageContents](#)
[EmitPrologString](#)
[EmitPSFontBegin](#)
[EmitPSFontEncodingBegin](#)
[EmitPSFontEncodingEnd](#)
[EmitPSFontEnd](#)
[EmitPSResourceBegin](#)
[EmitPSResourceEnd](#)
[EndSetup](#)
[FlushString](#)
[GetFontVMUsage](#)
[NotifyNewPage](#)
[PageBegin](#)
[PageEnd](#)
[PageSetup](#)
[PDPrintCanEmitFontProc](#)
[PDPrintEmitFontProc](#)
[PDPrintEmitPrologResourceProc](#)
[PDPrintGetFontEncodingMethodProc](#)

Data

The following data structures were added in version 4.0.

[AGMBlackPointFlt](#)
[AGMColorRangeFlt](#)
[AGMGrayCalFlt](#)
[AGMLabCalFlt](#)
[AGMRGBCalFlt](#)
[AGMWhitePointFlt](#)
[AGMXYZColorFlt](#)
[AVAnnotHandler](#)
[AVAnnotHandlerInfo](#)
[AVDocSaveParams](#)
[AVIcon](#)
[AVPageViewControlID](#)
[PDAnnotHandler](#)
[PDAnnotInfo](#)
[PDDocOpenParams](#)
[PDDocPreSaveInfo](#)
[PDEColorSpaceStruct](#)
[PDEDeviceNColorData](#)
[PDEFontCreateParams](#)
[PDEGrayCalFlt](#)
[PDEICCBasedColorData](#)
[PDEIndexedColorData](#)
[PDELabCalFlt](#)
[PDEPatternColorSpace](#)
[PDERGBCalFlt](#)
[PDESeparationColorData](#)
[PDFLPrintUserParamsRec](#) (for Adobe PDF Library)
[PDFontStyles](#)
[PDPageDrawFlags](#)
[PIHandshakeData_V0200](#)
[StdPassword](#)
[StdSecurityData](#)
[TKAllocatorProcs](#) (for Adobe PDF Library)
[TKResourceProcs](#) (for Adobe PDF Library)

The following data structures were added for Placed PDF.

[AGMALABColorRec](#)
[AGMARGBColorRec](#)
[AGMCMYKColorRec](#)
[AGMColorTab](#)
[AGMFixedMatrix](#)
[AGMFixedPoint](#)
[AGMImageAlphaRecord](#)
[AGMInt16Rect](#)
[AGMLABColorRec](#)
[AGMMemObj](#)
[AGMRGBColorRec](#)
[PDPageDrawFlags](#)
[PDPrintClient](#)

Errors

The following errors were added in version 4.0.

[avErrActionExternal](#)
[avErrActionFullScreen](#)
[avErrActionRestricted](#)
[avErrCantOpenDialog](#)
[avErrCantOpenPrinting](#)
[pageErrBadEPSColorSpace](#)
[pageErrBadFunction](#)
[pageErrInvalidImageMaskDepth](#)
[pdErrCannotMergeWithSubsetFonts](#)
[pdErrHostEncodingNotSet](#)
[pdErrNoPDDocForCosDoc](#)
[peErrBadBlockHeader](#)
[peErrCantCreateFontSubset](#)
[peErrCantEmbedFont](#)
[peErrCantGetAttrs](#)
[peErrCantGetWidths](#)
[peErrCantReadImage](#)
[peErrFontToEmbedNotOnSys](#)
[peErrNoError](#)
[peErrPStackUnderflow](#)
[peErrUnknownPDEColorSpace](#)
[peErrUnknownResType](#)
[peErrWrongPDEObjectType](#)

Notifications

The following notifications were added in version 4.0.

[AVAppWillCloseAllInternalDocs](#)

[AVDocDidClickName](#)

[AVDocDidPrint](#)

[AVDocWillPrint](#)

[PDBookmarkDidUnlink](#)

[PDDocDidClose](#)

[PDDocDidExportAnnots](#)

[PDDocDidImportAnnots](#)

[PDDocDidInsertPagesEx](#)

[PDDocPageLabelDidChange](#)

[PDDocWillClose](#)

[PDDocWillExportAnnots](#)

[PDDocWillImportAnnots](#)

[PDDocWillInsertPagesEx](#)

[PDNameTreeNameAdded](#)

[PDNameTreeNameRemoved](#)

[PSPrintAfterBeginPageSetup](#)

[PSPrintAfterBeginProlog](#)

[PSPrintAfterBeginSetup](#)

[PSPrintAfterEmitExtGState](#)

[PSPrintAfterPageTrailer](#)

[PSPrintAfterTrailer](#)

[PSPrintBeforeEndComments](#)

[PSPrintBeforeEndSetup](#)

A

AGMCleanup [1588](#)
 AGMClip [1589](#)
 AGMClosePath [1590](#)
 AGMConcat [1591](#)
 AGMDeletePort [1593](#)
 AGMDeleteRasterDev [1594](#)
 AGMFill [1595](#)
 AGMInit [1596](#)
 AGMLineTo [1597](#)
 AGMMoveTo [1598](#)
 AGMNewBitmapPort [1599](#)
 AGMNewPath [1601](#)
 AGMNewRasterDev [1602](#)
 AGMNewRasterPort [1604](#)
 AGMNewWindowPort [1606](#)
 AGMSetAntiAliasPolicy [1608](#)
 AGMSetRGBColor [1610](#)
 ASAtomExistsForString [11](#)
 ASAtomFromString [12](#)
 ASAtomGetCount [1575](#)
 ASAtomGetString [14](#)
 ASCallbackCreate [16](#)
 ASCallbackDestroy [18](#)
 ASCStringToFixed [84](#)
 ASEnumExtensions [20](#)
 ASEExtensionGetFileName [21](#)
 ASEExtensionGetRegisteredName [22](#)
 ASEExtensionMgrGetHFT [98](#)
 ASFileAcquirePathName [24](#)
 ASFileClose [25](#)
 ASFileFlush [26](#)
 ASFileFromMDFile [27](#)
 ASFileGetEOF [28](#)
 ASFileGetFileSys [29](#)
 ASFileGetFileSysByName [42](#)
 ASFileGetMDFile [30](#)
 ASFileGetPos [31](#)
 ASFilePushData [32](#)
 ASFileRead [34](#)
 ASFileRegisterFileSys [43](#)
 ASFileReopen [35](#)
 ASFileSetEOF [36](#)
 ASFileSetMode [37](#)
 ASFileSetPos [39](#)
 ASFileStmRdOpen [61](#)
 ASFileStmWrOpen [63](#)
 ASFileSysAcquireFileSysPath [44](#)
 ASFileSysCopyPath [46](#)
 ASFileSysCreatePathName [47](#)
 ASFileSysDIPathFromPath [49](#)
 ASFileSysOpenFile [51](#)
 ASFileSysPathFromDIPath [53](#)
 ASFileSysReleasePath [55](#)
 ASFileSysRemoveFile [56](#)
 ASFileUnregisterFileSys [57](#)
 ASFileWrite [40](#)
 ASFixedDiv [85](#)
 ASFixedMatrixConcat [86](#)
 ASFixedMatrixInvert [88](#)
 ASFixedMatrixTransform [90](#)
 ASFixedMatrixTransformRect [92](#)
 ASFixedMul [94](#)
 ASFixedToCString [95](#)
 ASfree [110](#)
 ASGetConfiguration [75](#)
 ASGetDefaultFileSys [58](#)
 ASGetErrorString [78](#)
 ASGetExceptionErrorCode [79](#)
 ASmalloc [111](#)
 ASMemStmRdOpen [65](#)
 ASPathFromPlatformPath [59](#)
 ASProcStmRdOpen [66](#)
 ASProcStmWrOpen [67](#)
 ASPurgeMemory [1576](#)
 ASRaise [80](#)
 ASrealloc [112](#)
 ASRegisterErrorString [81](#)
 ASStmClose [69](#)
 ASStmRead [70](#)
 ASStmWrite [72](#)
 AVActionHandlerGetProcs [116](#)
 AVActionHandlerGetType [118](#)
 AVActionHandlerGetUIName [119](#)
 AVAlert [121](#)
 AVAlertConfirm [123](#)
 AVAlertNote [125](#)
 AVAnnotHandlerDeleteInfo [128](#)
 AVAnnotHandlerGetInfo [129](#)
 AVAppBeginFullScreen [131](#)
 AVAppBeginModal [133](#)
 AVAppCanQuit [135](#)
 AVAppDoingFullScreen [136](#)
 AVAppEndFullScreen [137](#)
 AVAppEndModal [138](#)
 AVAppEnumActionHandlers [139](#)
 AVAppEnumAnnotHandlers [141](#)

AVAppEnumDocs [142](#)
AVAppEnumSystemFonts [1521](#)
AVAppEnumTools [144](#)
AVAppEnumTransHandlers [146](#)
AVAppGetActionHandlerByType [147](#)
AVAppGetActiveDoc [148](#)
AVAppGetActiveTool [150](#)
AVAppGetAnnotHandlerByName [151](#)
AVAppGetCancelProc [152](#)
AVAppGetDefaultTool [154](#)
AVAppGetDocProgressMonitor [155](#)
AVAppGetLanguage [157](#)
AVAppGetLastActiveTool [158](#)
AVAppGetMenubar [159](#)
AVAppGetName [160](#)
AVAppGetNumDocs [162](#)
AVAppGetPreference [163](#)
AVAppGetToolBar [164](#)
AVAppGetToolByName [165](#)
AVAppGetTransHandlerByType [166](#)
AVAppGetVersion [167](#)
AVAppHandleAppleEvent [1522](#)
AVAppHandlePlatformEvent [168](#)
AVAppIsIdle [169](#)
AVAppModalWindowsOpen [171](#)
AVAppOpenHelpFile [172](#)
AVAppRegisterActionHandler [173](#)
AVAppRegisterAnnotHandler [175](#)
AVAppRegisterForPageViewAdjustCursor [177](#)
AVAppRegisterForPageViewClicks [179](#)
AVAppRegisterForPageViewDrawing [181](#)
AVAppRegisterIdleProc [183](#)
AVAppRegisterNotification [185](#)
AVAppRegisterTool [187](#)
AVAppRegisterTransHandler [189](#)
AVAppSetActiveTool [190](#)
AVAppSetPreference [192](#)
AVAppUnregisterForPageViewAdjustCursor [194](#)
AVAppUnregisterForPageViewClicks [195](#)
AVAppUnregisterForPageViewDrawing [196](#)
AVAppUnregisterIdleProc [197](#)
AVAppUnregisterNotification [198](#)
AVAppWindowHandlePlatformEvent [1524](#)
AVAuthOpen [205](#)
AVCryptDoStdSecurity [206](#)
AVCryptGetPassword [207](#)
AVDestInfoDestroy [114](#)
AVDocAlert [210](#)
AVDocAlertConfirm [212](#)
AVDocAlertNote [213](#)
AVDocAlertYesNo [214](#)
AVDocClearSelection [216](#)
AVDocClose [218](#)
AVDocCopyAction [220](#)
AVDocCopyActionCommon [221](#)
AVDocCopyAdditionalActions [223](#)
AVDocCopyAnnot [225](#)
AVDocCopyAnnotCommon [227](#)
AVDocCopySelection [229](#)
AVDocDeleteSelection [231](#)
AVDocDoActionPropsDialog [233](#)
AVDocDoCopyAs [234](#)
AVDocDoPrint [235](#)
AVDocDoSave [236](#)
AVDocDoSaveAs [238](#)
AVDocDoSaveAsWithParams [239](#)
AVDocDoSelectionProperties [240](#)
AVDocEnumSelection [241](#)
AVDocGetAVWindow [243](#)
AVDocGetClientName [244](#)
AVDocGetPageText [246](#)
AVDocGetPageView [248](#)
AVDocGetPDDoc [249](#)
AVDocGetSelection [250](#)
AVDocGetSelectionServerByType [252](#)
AVDocGetSelectionType [253](#)
AVDocGetSplitterPosition [254](#)
AVDocGetViewDef [255](#)
AVDocGetViewMode [256](#)
AVDocIsExternal [257](#)
AVDocIsReadOnly [258](#)
AVDocOpenFromASFileWithParams [259](#)
AVDocOpenFromFile [260](#)
AVDocOpenFromFileWithParams [262](#)
AVDocOpenFromPDDoc [264](#)
AVDocOpenFromPDDocWithParams [266](#)
AVDocPerformAction [268](#)
AVDocPrintPages [270](#)
AVDocPrintPagesWithParams [272](#)
AVDocRegisterSelectionServer [275](#)
AVDocSelectionEnumPageRanges [278](#)
AVDocSendAuxData [280](#)
AVDocSetClientName [282](#)
AVDocSetDead [283](#)
AVDocSetReadOnly [284](#)
AVDocSetSelection [285](#)
AVDocSetSplitterPosition [289](#)

AVDocSetViewDef [290](#)
AVDocSetViewMode [291](#)
AVDocShowSelection [292](#)
AVExtensionMgrRegisterNotification [1577](#)
AVExtensionMgrUnregisterNotification [1579](#)
AVGrafSelectCreate [295](#)
AVGrafSelectDestroy [296](#)
AVGrafSelectGetBoundingRect [297](#)
AVHasAuxDataHandler [200](#)
AVMenuAcquire [299](#)
AVMenuAcquireMenuItemByIndex [301](#)
AVMenuAddMenuItem [303](#)
AVMenuBarAcquireMenuByIndex [322](#)
AVMenuBarAcquireMenuByName [324](#)
AVMenuBarAcquireMenuByPredicate [326](#)
AVMenuBarAcquireMenuItemByName [328](#)
AVMenuBarAcquireMenuItemByPredicate [330](#)
AVMenuBarAddHiddenMenu [332](#)
AVMenuBarAddMenu [333](#)
AVMenuBarGetMenuItemIndex [335](#)
AVMenuBarGetNumMenus [336](#)
AVMenuBarHide [337](#)
AVMenuBarShow [338](#)
AVMenuGetMenuItemIndex [305](#)
AVMenuGetName [307](#)
AVMenuGetNumMenuItems [309](#)
AVMenuGetParentMenuBar [310](#)
AVMenuGetParentMenuItem [311](#)
AVMenuGetTitle [312](#)
AVMenuIsHiddenOnMenuBar [314](#)
AVMenuItemAcquire [340](#)
AVMenuItemAcquireSubmenu [342](#)
AVMenuItemExecute [344](#)
AVMenuItemGetLongOnly [345](#)
AVMenuItemGetName [346](#)
AVMenuItemGetParentMenu [348](#)
AVMenuItemGetShortcut [349](#)
AVMenuItemGetTitle [351](#)
AVMenuItemIsEnabled [353](#)
AVMenuItemIsMarked [354](#)
AVMenuItemNew [355](#)
AVMenuItemRelease [358](#)
AVMenuItemRemove [360](#)
AVMenuItemSetComputeEnabledProc [362](#)
AVMenuItemSetComputeMarkedProc [364](#)
AVMenuItemSetExecuteProc [366](#)
AVMenuItemSetTitle [368](#)
AVMenuNew [315](#)
AVMenuRelease [317](#)
AVMenuRemove [319](#)
AVPageViewAcquireMachinePort [371](#)
AVPageViewAppearanceGetAVMatrix [372](#)
AVPageViewBeginOperation [374](#)
AVPageViewDevicePointToPage [375](#)
AVPageViewDeviceRectToPage [377](#)
AVPageViewDeviceRectToPageRZ [378](#)
AVPageViewDoPopupMenu [380](#)
AVPageViewDragOutNewRect [382](#)
AVPageViewDragRect [384](#)
AVPageViewDrawAnnotSequence [386](#)
AVPageViewDrawCosObj [387](#)
AVPageViewDrawCosObjEx [388](#)
AVPageViewDrawNow [390](#)
AVPageViewDrawRect [392](#)
AVPageViewDrawRectOutline [394](#)
AVPageViewEndOperation [396](#)
AVPageViewGetActiveBead [397](#)
AVPageViewGetAnnotRect [398](#)
AVPageViewGetAperture [400](#)
AVPageViewGetAVDoc [401](#)
AVPageViewGetColor [402](#)
AVPageViewGetDevToPageMatrix [403](#)
AVPageViewGetFirstVisiblePageNum [404](#)
AVPageViewGetGrayRect [405](#)
AVPageViewGetLastVisiblePageNum [406](#)
AVPageViewGetLayoutMode [407](#)
AVPageViewGetMousePosition [408](#)
AVPageViewGetNextView [410](#)
AVPageViewGetPage [412](#)
AVPageViewGetPageNum [413](#)
AVPageViewGetPageToDevMatrix [414](#)
AVPageViewGetSelectedAnnotPageNum [415](#)
AVPageViewGetThreadIndex [416](#)
AVPageViewGetVisibleAnnotPage [417](#)
AVPageViewGetZoom [418](#)
AVPageViewGetZoomType [419](#)
AVPageViewGoBack [420](#)
AVPageViewGoForward [421](#)
AVPageViewGoTo [422](#)
AVPageViewHighlightText [423](#)
AVPageViewInsetRect [424](#)
AVPageViewInvalidateRect [425](#)
AVPageViewInvalidateText [427](#)
AVPageViewInvertQuad [428](#)
AVPageViewInvertRect [429](#)
AVPageViewInvertRectOutline [431](#)
AVPageViewIsAnnotAtPoint [433](#)

AVPageViewsBeadAtPoint [435](#)
AVPageViewPageNumIsVisible [436](#)
AVPageViewPointInText [437](#)
AVPageViewPointToDevice [438](#)
AVPageViewReadPageDown [440](#)
AVPageViewReadPageUp [441](#)
AVPageViewRectToDevice [442](#)
AVPageViewReleaseMachinePort [443](#)
AVPageViewScrollTo [444](#)
AVPageViewScrollToRect [445](#)
AVPageViewSetAnnotLocation [447](#)
AVPageViewSetColor [449](#)
AVPageViewSetLayoutMode [451](#)
AVPageViewSetPageNum [452](#)
AVPageViewShowControl [454](#)
AVPageViewStartReadingThread [455](#)
AVPageViewToDestInfo [456](#)
AVPageViewToViewDest [457](#)
AVPageViewTrackText [458](#)
AVPageViewTransformRectRZ [459](#)
AVPageViewUseDestInfo [461](#)
AVPageViewUseThisDestination [462](#)
AVPageViewZoomTo [464](#)
AVRectToRect [1525](#)
AVRegisterAuxDataHandler [201](#)
AVSysAllocTimeStringFromTimeRec [467](#)
AVSysBeep [468](#)
AVSysGetCursor [469](#)
AVSysGetModifiers [470](#)
AVSysGetStandardCursor [471](#)
AVSysMouselsStillDown [472](#)
AVSysSetCursor [474](#)
AVToolBarAddButton [480](#)
AVToolBarEnumButtons [482](#)
AVToolBarGetButtonByName [484](#)
AVToolBarGetFrame [485](#)
AVToolBarGetNumButtons [486](#)
AVToolBarIsRoomFor [487](#)
AVToolBarNewFlyout [489](#)
AVToolBarUpdateButtonStates [490](#)
AVToolButtonDestroy [492](#)
AVToolButtonExecute [493](#)
AVToolButtonGetFlyout [494](#)
AVToolButtonGetIcon [495](#)
AVToolButtonGetMenu [496](#)
AVToolButtonGetName [497](#)
AVToolButtonIsEnabled [498](#)
AVToolButtonIsMarked [499](#)
AVToolButtonIsSeparator [500](#)
AVToolButtonNew [501](#)
AVToolButtonRemove [503](#)
AVToolButtonSetComputeEnabledProc [504](#)
AVToolButtonSetComputeMarkedProc [506](#)
AVToolButtonSetExecuteProc [508](#)
AVToolButtonSetExternal [510](#)
AVToolButtonSetFlyout [512](#)
AVToolButtonSetHelpText [513](#)
AVToolButtonSetIcon [515](#)
AVToolButtonSetMenu [516](#)
AVToolGetType [477](#)
AVToolsIsPersistent [478](#)
AVUnregisterAuxDataHandler [203](#)
AVWindowBecomeKey [518](#)
AVWindowBringToFront [519](#)
AVWindowDestroy [520](#)
AVWindowDrawNow [521](#)
AVWindowGetCursorAtPoint [1526](#)
AVWindowGetFrame [522](#)
AVWindowGetInterior [524](#)
AVWindowGetOwnerData [525](#)
AVWindowGetPlatformThing [526](#)
AVWindowGetTitle [527](#)
AVWindowHandlePlatformEvent [528](#)
AVWindowHide [529](#)
AVWindowInvalidateRect [530](#)
AVWindowIsKey [531](#)
AVWindowIsVisible [532](#)
AVWindowMaximize [533](#)
AVWindowNew [534](#)
AVWindowNewFromPlatformThing [536](#)
AVWindowResignKey [538](#)
AVWindowSetFrame [539](#)
AVWindowSetOwnerData [540](#)
AVWindowSetTitle [541](#)
AVWindowSetWantsKey [543](#)
AVWindowShow [542](#)
AVWindowUserClose [545](#)

C

CCSetSystemCalibration [1612](#)
CosArrayGet [547](#)
CosArrayInsert [549](#)
CosArrayLength [550](#)
CosArrayPut [551](#)
CosArrayRemove [553](#)
CosArrayRemoveNth [555](#)
CosBooleanValue [559](#)

CosDecryptData [630](#)
CosDictGet [562](#)
CosDictKnown [564](#)
CosDictPut [566](#)
CosDictRemove [568](#)
CosDocClose [573](#)
CosDocCreate [574](#)
CosDocEnumEOFs [575](#)
CosDocEnumIndirect [577](#)
CosDocGetInfoDict [579](#)
CosDocGetObjByID [580](#)
CosDocGetRoot [581](#)
CosDocOpenWithParams [582](#)
CosDocSaveToFile [584](#)
CosDocSaveWithParams [586](#)
CosDocSetDirty [588](#)
CosEncryptData [632](#)
CosFixedValue [590](#)
CosIntegerValue [594](#)
CosNameValue [597](#)
CosNewArray [556](#)
CosNewBoolean [560](#)
CosNewDict [570](#)
CosNewFixed [591](#)
CosNewInteger [595](#)
CosNewName [598](#)
CosNewNull [600](#)
CosNewStream [614](#)
CosNewString [624](#)
CosObjCopy [602](#)
CosObjDestroy [603](#)
CosObjEnum [604](#)
CosObjEqual [606](#)
CosObjGetDoc [607](#)
CosObjGetGeneration [608](#)
CosObjGetID [609](#)
CosObjGetType [610](#)
CosObjHash [611](#)
CosObjIsIndirect [612](#)
CosStreamDict [618](#)
CosStreamLength [619](#)
CosStreamOpenStm [620](#)
CosStreamPos [621](#)
CosStringGetHexFlag [626](#)
CosStringSetHexFlag [627](#)
CosStringValue [628](#)

H

HFTDestroy [99](#)
HFTGetReplacedEntry [100](#)
HFTIsValid [102](#)
HFTNew [103](#)
HFTReplaceEntry [104](#)
HFTServerDestroy [106](#)
HFTServerNew [107](#)

P

PDActionDestroy [669](#)
PDActionEqual [670](#)
PDActionFromCosObj [671](#)
PDActionGetCosObj [672](#)
PDActionGetDest [673](#)
PDActionGetFileSpec [675](#)
PDActionGetSubtype [676](#)
PDActionIsValid [677](#)
PDActionNew [678](#)
PDActionNewFromDest [679](#)
PDActionNewFromFileSpec [681](#)
PDAnnotEqual [683](#)
PDAnnotFromCosObj [684](#)
PDAnnotGetColor [685](#)
PDAnnotGetCosObj [687](#)
PDAnnotGetDate [689](#)
PDAnnotGetFlags [690](#)
PDAnnotGetRect [691](#)
PDAnnotGetSubtype [693](#)
PDAnnotGetTitle [695](#)
PDAnnotIsValid [697](#)
PDAnnotNotifyDidChange [699](#)
PDAnnotNotifyWillChange [701](#)
PDAnnotSetColor [702](#)
PDAnnotSetDate [704](#)
PDAnnotSetFlags [705](#)
PDAnnotSetRect [706](#)
PDAnnotSetTitle [708](#)
PDBeadAcquirePage [714](#)
PDBeadDestroy [715](#)
PDBeadEqual [716](#)
PDBeadFromCosObj [718](#)
PDBeadGetCosObj [719](#)
PDBeadGetIndex [720](#)
PDBeadGetNext [721](#)
PDBeadGetPrev [723](#)
PDBeadGetRect [725](#)

PDBeadGetThread [726](#)
PDBeadInsert [727](#)
PDBeadIsValid [728](#)
PDBeadNew [730](#)
PDBeadSetPage [731](#)
PDBeadSetRect [732](#)
PDBookmarkAddChild [734](#)
PDBookmarkAddNewChild [736](#)
PDBookmarkAddNewSibling [737](#)
PDBookmarkAddNext [738](#)
PDBookmarkAddPrev [739](#)
PDBookmarkAddSubtree [740](#)
PDBookmarkDestroy [742](#)
PDBookmarkEqual [743](#)
PDBookmarkFromCosObj [744](#)
PDBookmarkGetAction [745](#)
PDBookmarkGetByTitle [746](#)
PDBookmarkGetCosObj [748](#)
PDBookmarkGetCount [749](#)
PDBookmarkGetFirstChild [750](#)
PDBookmarkGetIndent [751](#)
PDBookmarkGetLastChild [752](#)
PDBookmarkGetNext [753](#)
PDBookmarkGetParent [754](#)
PDBookmarkGetPrev [755](#)
PDBookmarkGetTitle [756](#)
PDBookmarkHasChildren [758](#)
PDBookmarksOpen [759](#)
PDBookmarksIsValid [760](#)
PDBookmarkSetAction [761](#)
PDBookmarkSetOpen [762](#)
PDBookmarkSetTitle [763](#)
PDBookmarkUnlink [765](#)
PDCharProcEnum [767](#)
PDCharProcGetCosObj [769](#)
PDDocAcquire [771](#)
PDDocAcquirePage [772](#)
PDDocAddThread [773](#)
PDDocAuthorize [774](#)
PDDocClearFlags [776](#)
PDDocClose [777](#)
PDDocCreate [778](#)
PDDocCreateNameTree [779](#)
PDDocCreatePage [781](#)
PDDocCreateStructTreeRoot [783](#)
PDDocCreateTextSelect [785](#)
PDDocCreateThumbs [787](#)
PDDocCreateWordFinder [790](#), [794](#)
PDDocCreateWordFinderUCS [794](#)
PDDocDeletePages [797](#)
PDDocDeleteThumbs [799](#)
PDDocEnumFonts [801](#)
PDDocEnumLoadedFonts [803](#)
PDDocEnumResources [804](#)
PDDocExportNotes [806](#)
PDDocFindPageNumForLabel [808](#)
PDDocFromCosDoc [810](#)
PDDocGetBookmarkRoot [811](#)
PDDocGetCosDoc [812](#)
PDDocGetCryptHandlerClientData [813](#)
PDDocGetFile [814](#)
PDDocGetFlags [815](#)
PDDocGetFullScreen [816](#)
PDDocGetID [817](#)
PDDocGetInfo [819](#)
PDDocGetLabelForPageNum [821](#)
PDDocGetNameTree [823](#)
PDDocGetNewCryptHandler [824](#)
PDDocGetNewSecurityData [825](#)
PDDocGetNewSecurityInfo [826](#)
PDDocGetNumPages [828](#)
PDDocGetNumThreads [829](#)
PDDocGetOpenAction [831](#)
PDDocGetPageLabel [832](#)
PDDocGetPageMode [834](#)
PDDocGetPermissions [835](#)
PDDocGetSecurityData [836](#)
PDDocGetStructTreeRoot [837](#)
PDDocGetThread [838](#)
PDDocGetThreadIndex [840](#)
PDDocGetVersion [841](#)
PDDocGetWordFinder [842](#)
PDDocImportCosDocNotes [844](#)
PDDocImportNotes [846](#)
PDDocInsertPages [848](#)
PDDocMovePage [852](#)
PDDocNewSecurityData [854](#)
PDDocOpen [855](#)
PDDocOpenEx [858](#)
PDDocOpenFromASFile [861](#)
PDDocOpenFromASFileEx [863](#)
PDDocOpenWithParams [865](#)
PDDocPrintPages [1613](#)
PDDocReadAhead [866](#)
PDDocReadAheadPages [867](#)
PDDocRelease [868](#)
PDDocRemoveNameTree [869](#)
PDDocRemovePageLabel [870](#)

PDDocRemoveStructTreeRoot [871](#)
PDDocRemoveThread [872](#)
PDDocReplacePages [873](#)
PDDocSave [876](#)
PDDocSaveWithParams [879](#)
PDDocSetFlags [880](#)
PDDocSetFullScreen [881](#)
PDDocSetInfo [882](#)
PDDocSetNewCryptHandler [884](#)
PDDocSetNewSecurityData [886](#)
PDDocSetOpenAction [888](#)
PDDocSetPageLabel [889](#)
PDDocSetPageMode [891](#)
PDDrawCosObjToWindow [635](#)
PDEAcquire [1334](#)
PDEAddTag [1335](#)
PDEAttrEnumTable [1207](#)
PDEClipAddElem [1216](#)
PDEClipCopy [1217](#)
PDEClipCreate [1218](#)
PDEClipFlattenedEnumElems [1219](#)
PDEClipGetElem [1221](#)
PDEClipGetNumElems [1222](#)
PDEClipRemoveElems [1223](#)
PDEColorSpaceCreate [1225](#)
PDEColorSpaceCreateFromCosObj [1225](#)
PDEColorSpaceCreateFromName [1227](#)
PDEColorSpaceGetBase [1228](#)
PDEColorSpaceGetBaseNumComps [1229](#)
PDEColorSpaceGetCosObj [1230](#)
PDEColorSpaceGetCTable [1232](#)
PDEColorSpaceGetHiVal [1234](#)
PDEColorSpaceGetName [1235](#)
PDEColorSpaceGetNumComps [1236](#)
PDEContainerCreate [1238](#)
PDEContainerGetContent [1239](#)
PDEContainerGetDict [1240](#)
PDEContainerGetMCTag [1241](#)
PDEContainerSetContent [1242](#)
PDEContainerSetDict [1243](#)
PDEContainerSetMCTag [1244](#)
PDEContentAddElem [1246](#)
PDEContentCreate [1247](#)
PDEContentCreateFromCosObj [1248](#)
PDEContentGetAttrs [1250](#)
PDEContentGetDefaultColorSpace [1251](#)
PDEContentGetElem [1252](#)
PDEContentGetNumElems [1253](#)
PDEContentGetResources [1254](#)
PDEContentRemoveElem [1256](#)
PDEContentToCosObj [1257](#)
PDEDefaultGState [1209](#)
PDEDeviceNColorsCreate [1261](#)
PDEDeviceNColorsGetColorValue [1262](#)
PDEElementCopy [1264](#)
PDEElementGetBBox [1265](#)
PDEElementGetClip [1266](#)
PDEElementGetGState [1267](#)
PDEElementGetMatrix [1269](#)
PDEElementIsAtPoint [1271](#)
PDEElementIsAtRect [1272](#)
PDEElementSetClip [1272](#)
PDEElementSetGState [1274](#)
PDEElementSetMatrix [1276](#)
PDEEnumElements [1210](#)
PDEExtGStateCreate [1278](#)
PDEExtGStateGetCosObj [1279](#)
PDEFontCreate [1281](#)
PDEFontCreateFromCosObj [1284](#)
PDEFontCreateFromSysFont [1285](#)
PDEFontCreateFromSysFontEx [1287](#)
PDEFontCreateWithParams [1289](#)
PDEFontGetAttrs [1289](#)
PDEFontGetCosObj [1292](#)
PDEFontGetNumCodeBytes [1293](#)
PDEFontGetOneByteEncoding [1294](#)
PDEFontGetWidths [1293](#)
PDEFontIsMultiByte [1296](#)
PDEFontSubsetNow [1297](#)
PDEFontSumWidths [1299](#)
PDEFormCreateFromCosObj [1303](#)
PDEFormGetContent [1304](#)
PDEFormGetCosObj [1305](#)
PDEGetTag [1337](#)
PDEGroupCreate [1307](#)
PDEGroupGetContent [1308](#)
PDEGroupSetContent [1309](#)
PDEImageCreate [1311](#)
PDEImageCreateFromCosObj [1313](#)
PDEImageDataIsEncoded [1315](#)
PDEImageGetAttrs [1317](#)
PDEImageGetColorSpace [1318](#)
PDEImageGetCosObj [1320](#)
PDEImageGetData [1321](#)
PDEImageGetDataLen [1323](#)
PDEImageGetDataStm [1324](#)
PDEImageGetFilterArray [1326](#)
PDEImageIsCosObj [1327](#)

[PDEImageSetData 1328](#)
[PDEImageSetDataStm 1330](#)
[PDELogDump 1204](#)
[PDEmbedSysFontForPDFont 1412](#)
[PDEMergeResourcesDict 1212](#)
[PDEnumDocs 638](#)
[PDEnumSysFonts 1414](#)
[PDEObjectDump 1205](#)
[PDEObjectGetType 1338](#)
[PDEPathAddSegment 1342](#)
[PDEPathCreate 1344](#)
[PDEPathGetData 1345](#)
[PDEPathGetPaintOp 1347](#)
[PDEPathSetData 1348](#)
[PDEPathSetPaintOp 1350](#)
[PDEPatternCreate 1352](#)
[PDEPatternGetCosObj 1353](#)
[PDEPlaceCreate 1355](#)
[PDEPlaceGetDict 1356](#)
[PDEPlaceGetMCTag 1357](#)
[PDEPlaceSetDict 1358](#)
[PDEPlaceSetMCTag 1359](#)
[PDEPurgeCache 1214](#)
[PDERelease 1339](#)
[PDERemoveTag 1340](#)
[PDEShadingCreateFromCosObj 1361](#)
[PDEShadingGetCosObj 1362](#)
[PDETextAdd 1364](#)
[PDETextCreate 1366](#)
[PDETextGetAdvanceWidth 1367](#)
[PDETextGetBBox 1369](#)
[PDETextGetFont 1371](#)
[PDETextGetGState 1373](#)
[PDETextGetNumBytes 1375](#)
[PDETextGetNumChars 1375](#)
[PDETextGetNumRuns 1378](#)
[PDETextGetQuad 1379](#)
[PDETextGetRunForChar 1381](#)
[PDETextGetStrokeMatrix 1382](#)
[PDETextGetText 1384](#)
[PDETextGetTextMatrix 1386](#)
[PDETextGetTextState 1388](#)
[PDETextIsAtPoint 1390](#)
[PDETextIsAtRect 1392](#)
[PDETextRemove 1390](#)
[PDETextReplaceChars 1396](#)
[PDETextRunGetCharOffset 1398](#)
[PDETextRunGetNumChars 1399](#)
[PDETextRunSetFont 1400](#)
[PDETextRunSetGState 1401](#)
[PDETextRunSetStrokeMatrix 1402](#)
[PDETextRunSetTextMatrix 1403](#)
[PDETextRunSetTextState 1404](#)
[PDETextSplitRunAt 1405](#)
[PDEUnknownGetOpName 1407](#)
[PDEXObjectCreate 1409](#)
[PDEXObjectGetCosObj 1410](#)
[PDFFileSpecAcquireASPath 893](#)
[PDFFileSpecFromCosObj 894](#)
[PDFFileSpecGetCosObj 895](#)
[PDFFileSpecGetDIPath 896](#)
[PDFFileSpecGetDoc 898](#)
[PDFFileSpecGetFileSys 899](#)
[PDFFileSpecGetFileSysName 900](#)
[PDFFileSpecIsValid 901](#)
[PDFFileSpecNewFromASPath 902](#)
[PDFindSysFont 1416](#)
[PDFindSysFontEx 1418](#)
[PDFindSysFontForPDFont 1300](#)
[PDFLGetVersion 1581](#)
[PDFLInit 1582](#)
[PDFLPrintDoc 1584](#)
[PDFLPrintPDF 1584](#)
[PDFLTerm 1586](#)
[PDFontAcquireEncodingArray 905](#)
[PDFontAcquireXlateTable 907](#)
[PDFontDownloadContextCreate 1614](#)
[PDFontDownloadContextDestroy 1615](#)
[PDFontEncodingArrayRelease 908](#)
[PDFontEnumCharProcs 909](#)
[PDFontFromCosObj 911](#)
[PDFontGetBBox 912](#)
[PDFontGetCharSet 913](#)
[PDFontGetCIDSystemInfo 914](#)
[PDFontGetCIDSystemSupplement 916](#)
[PDFontGetCosObj 918](#)
[PDFontGetDescendant 919](#)
[PDFontGetEncodingIndex 920](#)
[PDFontGetEncodingName 921](#)
[PDFontGetFontMatrix 923](#)
[PDFontGetMetrics 924](#)
[PDFontGetName 926](#)
[PDFontGetSubtype 928](#)
[PDFontGetWidths 929](#)
[PDFontIsEmbedded 931](#)
[PDFontSetMetrics 932](#)
[PDFontStreamPS 1616](#)
[PDFontWasExtracted 1617](#)

PDFontWasFauxed [1618](#)
PDFontXlateString [934](#)
PDFontXlateTableRelease [936](#)
PDFontXlateToHost [937](#)
PDFontXlateToUCS [940](#)
PDFontXlateWidths [943](#)
PDFormEnumPaintProc [946](#)
PDFormEnumResources [947](#)
PDFormGetBBox [948](#)
PDFormGetFormType [949](#)
PDFormGetMatrix [950](#)
PDFormGetXUIDCosObj [951](#)
PDFreeMemory [1619](#)
PDGetAnnotHandlerByName [711](#)
PDGetHostEncoding [639](#)
PDGetPDFDocEncoding [641](#)
PDGraphicGetBBox [953](#)
PDGraphicGetCurrentMatrix [954](#)
PDGraphicGetState [955](#)
PDHostMBLen [643](#)
PDImageColorSpaceGetIndexLookup [957](#)
PDImageGetAttrs [959](#)
PDImageSelAdjustMatrix [962](#)
PDImageSelectAlternate [960](#)
PDImageSelGetDeviceAttr [964](#)
PDImageSelGetGeoAttr [966](#)
PDInlinelImageColorSpaceGetIndexLookup [969](#)
PDInlinelImageGetAttrs [971](#)
PDInlinelImageGetData [973](#)
PDLinkAnnotGetAction [976](#)
PDLinkAnnotGetBorder [978](#)
PDLinkAnnotSetAction [979](#)
PDLinkAnnotSetBorder [981](#)
PDNameTreeEnum [984](#)
PDNameTreeEqual [985](#)
PDNameTreeFromCosObj [986](#)
PDNameTreeGet [987](#)
PDNameTreeGetCosObj [988](#)
PDNameTreeValid [989](#)
PDNameTreeLookup [990](#)
PDNameTreeNew [992](#)
PDNameTreePut [993](#)
PDNameTreeRemove [994](#)
PDNumTreeEnum [996](#)
PDNumTreeEqual [997](#)
PDNumTreeFromCosObj [998](#)
PDNumTreeGet [999](#)
PDNumTreeGetCosObj [1000](#)
PDNumTreeValid [1001](#)
PDNumTreeNew [1002](#)
PDNumTreePut [1003](#)
PDNumTreeRemove [1004](#)
PDPageAcquirePDEContent [1006](#)
PDPageAddAnnot [1008](#)
PDPageAddCosContents [1010](#)
PDPageAddCosResource [1011](#)
PDPageAddNewAnnot [1013](#)
PDPageCreateAnnot [1015](#)
PDPageDrawContents [1620](#)
PDPageDrawContentsPlaced [1622](#)
PDPageDrawContentsToWindow [1017](#)
PDPageEmitPSOrient [1626](#)
PDPageEnumContents [1021](#)
PDPageEnumResources [1023](#)
PDPageGetAnnot [1025](#)
PDPageGetAnnotIndex [1028](#)
PDPageGetAnnotSequence [1029](#)
PDPageGetBBox [1030](#)
PDPageGetCosObj [1031](#)
PDPageGetCosResources [1032](#)
PDPageGetCropBox [1033](#)
PDPageGetDefaultMatrix [1034](#)
PDPageGetDoc [1035](#)
PDPageGetDuration [1036](#)
PDPageGetFlippedMatrix [1037](#)
PDPageGetMediaBox [1038](#)
PDPageGetNumAnnots [1039](#)
PDPageGetNumber [1040](#)
PDPageGetPDEContentFilters [1041](#)
PDPageGetPDEContentFlags [1043](#)
PDPageGetRotate [1044](#)
PDPageGetTransition [1045](#)
PDPageHasTransition [1046](#)
PDPageLabelEqual [1083](#)
PDPageLabelFromCosObj [1084](#)
PDPageLabelGetCosObj [1085](#)
PDPageLabelGetPrefix [1086](#)
PDPageLabelGetStart [1087](#)
PDPageLabelGetStyle [1088](#)
PDPageLabellsValid [1089](#)
PDPageLabelNew [1090](#)
PDPageNotifyContentsDidChange [1047](#)
PDPageNotifyContentsDidChangeEx [1049](#)
PDPageNumFromCosObj [1051](#)
PDPagePDEContentWasChanged [1052](#)
PDPageRegisterForPDEContentChanged [1054](#)

PDPageRegisterForPDEContentNotCached [1056](#)
PDPageRelease [1058](#)
PDPageReleasePDEContent [1059](#)
PDPageRemoveAnnot [1061](#)
PDPageRemoveCosContents [1063](#)
PDPageRemoveCosResource [1064](#)
PDPageSetCropBox [1065](#)
PDPageSetDuration [1066](#)
PDPageSetMediaBox [1067](#)
PDPageSetPDEContent [1068](#)
PDPageSetPDEContentFilters [1070](#)
PDPageSetPDEContentFlags [1071](#)
PDPageSetRotate [1072](#)
PDPageSetTransition [1073](#)
PDPageStmGetInlinelImage [1074](#)
PDPageStmGetToken [1076](#)
PDPageUnRegisterForPDEContentChanged [1078](#)
PDPageUnRegisterForPDEContentNotCached [1080](#)
PDPathEnum [1093](#)
PDPathGetPaintOp [1095](#)
PDPrefGetColorCal [645](#)
PDPrefSetColorCal [647](#)
PDRegisterAnnotHandler [712](#)
PDRegisterCryptHandler [649](#)
PDRegisterCryptHandlerEx [652](#)
PDRegisterFileSpecHandler [655](#)
PDRegisterFileSpecHandlerByName [657](#)
PDSAttrObjCreate [1438](#)
PDSAttrObjCreateFromStream [1440](#)
PDSAttrObjGetOwner [1441](#)
PDSClassMapAddAttrObj [1443](#)
PDSClassMapGetAttrObj [1444](#)
PDSClassMapGetNumAttrObjs [1446](#)
PDSClassMapRemoveAttrObj [1447](#)
PDSClassMapRemoveClass [1448](#)
PDSElementAddAttrObj [1450](#)
PDSElementAddClass [1451](#)
PDSElementClearID [1452](#)
PDSElementCreate [1453](#)
PDSElementGetAlt [1454](#)
PDSElementGetAttrObj [1455](#)
PDSElementGetClass [1457](#)
PDSElementGetFirstPage [1459](#)
PDSElementGetID [1461](#)
PDSElementGetKid [1462](#)
PDSElementGetNumAttrObjs [1464](#)
PDSElementGetNumClasses [1465](#)
PDSElementGetNumKids [1466](#)
PDSElementGetParent [1467](#)
PDSElementGetRevision [1469](#)
PDSElementGetStructTreeRoot [1470](#)
PDSElementGetTitle [1471](#)
PDSElementGetType [1472](#)
PDSElementIncrementRevision [1473](#)
PDSElementInsertKid [1474](#)
PDSElementInsertMCAsKid [1475](#)
PDSElementInsertOBJAsKid [1477](#)
PDSElementRemoveAllAttrObjs [1478](#)
PDSElementRemoveAllClasses [1479](#)
PDSElementRemoveAttrObj [1480](#)
PDSElementRemoveClass [1482](#)
PDSElementRemoveKid [1484](#)
PDSElementRemoveKidMC [1485](#)
PDSElementReplaceKid [1486](#)
PDSElementReplaceKidMC [1487](#)
PDSElementReplaceKidOBJ [1489](#)
PDSElementSetAlt [1490](#)
PDSElementSetID [1491](#)
PDSElementSetTitle [1493](#)
PDSElementSetType [1494](#)
PDSetHostEncoding [659](#)
PDSMCGetParent [1496](#)
PDSElementGetParent [1498](#)
PDSRoleMapCopy [1500](#)
PDSRoleMapDoesMap [1501](#)
PDSRoleMapGetDirectMap [1502](#)
PDSRoleMapMap [1503](#)
PDSRoleMapUnMapDst [1504](#)
PDSRoleMapUnMapSrc [1505](#)
PDSTreeRootCreateClassMap [1507](#)
PDSTreeRootCreateRoleMap [1508](#)
PDSTreeRootGetClassMap [1509](#)
PDSTreeRootGetElementFromID [1510](#)
PDSTreeRootGetKid [1512](#)
PDSTreeRootGetNumKids [1513](#)
PDSTreeRootGetRoleMap [1514](#)
PDSTreeRootInsertKid [1515](#)
PDSTreeRootRemoveClassMap [1516](#)
PDSTreeRootRemoveKid [1517](#)
PDSTreeRootRemoveRoleMap [1518](#)
PDSTreeRootReplaceKid [1519](#)
PDStyleGetColor [1097](#)
PDStyleGetFont [1098](#)
PDStyleGetFontSize [1099](#)
PDSysFontAcquirePlatformData [1420](#)

PDSysFontGetAttrs [1420, 1421](#)
PDSysFontGetCIDSystemInfo [1423](#)
PDSysFontGetEncoding [1425](#)
PDSysFontGetInfo [1427](#)
PDSysFontGetName [1429](#)
PDSysFontGetType0Widths [1430](#)
PDSysFontGetWidths [1430, 1433](#)
PDSysFontGetWidthsEx [1434](#)
PDSysFontReleasePlatformData [1436](#)
PDTextAnnotGetContents [1104](#)
PDTextAnnotIsOpen [1106](#)
PDTextAnnotSetContents [1107](#)
PDTextAnnotSetOpen [1109](#)
PDTextEnum [1101](#)
PDTextGetState [1102](#)
PDTextSelectCreatePageHilite [1111](#)
PDTextSelectCreateRanges [1113](#)
PDTextSelectCreateWordHilite [1115](#)
PDTextSelectDestroy [1117](#)
PDTextSelectEnumQuads [1118](#)
PDTextSelectEnumText [1120](#)
PDTextSelectGetBoundingRect [1122](#)
PDTextSelectGetPage [1123](#)
PDTextSelectGetRange [1124](#)
PDTextSelectGetRangeCount [1125](#)
PDThreadDestroy [1127](#)
PDThreadFromCosObj [1128](#)
PDThreadGetCosObj [1129](#)
PDThreadGetFirstBead [1130](#)
PDThreadGetInfo [1131](#)
PDThreadIsValid [1133](#)
PDThreadNew [1134](#)
PDThreadSetFirstBead [1135](#)
PDThreadSetInfo [1136](#)
PDTransEqual [1139](#)
PDTransFromCosObj [1140](#)
PDTransGetCosObj [1141](#)
PDTransGetDuration [1142](#)
PDTransGetSubtype [1143](#)
PDTransIsValid [1144](#)
PDTransNew [1145](#)
PDTransNewFromCosDoc [1147](#)
PDTransNull [1148](#)
PDViewDestCreate [1150](#)
PDViewDestDestroy [1152](#)
PDViewDestFromCosObj [1153](#)
PDViewDestGetAttr [1154](#)
PDViewDestGetCosObj [1156](#)
PDViewDestIsValid [1157](#)

PDViewDestResolve [1158](#)
PDWordFilterString [1160](#)
PDWordFilterWord [1162](#)
PDWordFinderAcquireWordList [1186](#)
PDWordFinderDestroy [1190](#)
PDWordFinderEnumWords [1192](#)
PDWordFinderGetLatestAlgVersion [1194](#)
PDWordFinderGetNthWord [1195](#)
PDWordFinderReleaseWordList [1196](#)
PDWordGetAttr [1164](#)
PDWordGetCharacterTypes [1166](#)
PDWordGetCharDelta [1168](#)
PDWordGetCharOffset [1169](#)
PDWordGetLength [1171](#)
PDWordGetNthCharStyle [1173](#)
PDWordGetNthQuad [1175](#)
PDWordGetNumQuads [1177](#)
PDWordGetString [1178](#)
PDWordGetStyleTransition [1180](#)
PDWordsRotated [1182](#)
PDWordSplitString [1183](#)
PDXlateToHost [660, 662](#)
PDXlateToHostEx [662](#)
PDXlateToPDFDocEnc [664, 666](#)
PDXlateToPDFDocEncEx [666](#)
PDXObjectEnumFilters [1198](#)
PDXObjectGetCosObj [1199](#)
PDXObjectGetData [1200](#)
PDXObjectGetDataLength [1201](#)
PDXObjectGetSubtype [1202](#)

R

RectToAVRect [1527](#)

U

UnixAppAddModifierCallback [1529](#)
UnixAppClipboardGetItemId [1531](#)
UnixAppDispatchEvent [1532](#)
UnixAppGetAppShellWidget [1533](#)
UnixAppGetPlugInFilename [1536](#)
UnixAppLoadPlugInAppDefaults [1537](#)
UnixAppProcessEvent [1540](#)
UnixAppRemoveModifierCallback [1541](#)
UnixAppWaitForWm [1542](#)
UnixSysGetConfigName [1544](#)
UnixSysGetCursor [1545](#)
UnixSysGetCwd [1547](#)

UnixSysGetHomeDirectory [1548](#)
UnixSysGetHostname [1549](#)
UnixSysGetIcon [1550](#)
UnixSysGetInstallDirectory [1552](#)
UnixSysGetPixmap [1553](#)
UnixSysGetString [1556](#)
UnixSysGetTempFileDirectory [1558](#)
UnixSysPrefInit [1559](#)
UnixSysPrefUpdate [1565](#)

W

WinAppEnableIdleTimer [1568](#)
WinAppGetModalParent [1569](#)
WinAppGetPalette [1570](#)
WinAppGetPrinterHDC [1571](#)
WinAppRegisterModelessDialog [1572](#)
WinAppUnRegisterModelessDialog [1573](#)