Acrobat Viewer Interapplication Communication On-Line Reference

Adobe Developer Support

Technical Note #5165

1 May 1996

Adobe Systems Incorporated

Corporate Headquarters
1585 Charleston Road PO Box 7900
Mountain View, CA 94039-7900
(415) 961-4400 Main Number
(415) 961-4111 Developer Support
Fax: (415) 967-9231

European Engineering Support Group
Adobe Systems Benelux B.V.
P.O Box 22750
1100 DG Amsterdam, The Netherlands
+31-20-6511 275
Fax: +31-20-6511 313

Adobe Systems Eastern Region
24 New England
Executive Park
Burlington, MA 01803
(617) 273-2120
Fax: (617) 273-2336

Adobe Systems Japan
Swiss Bank House 7F
4-1-8 Toranomon, Minato-ku
Tokyo 105, Japan
+81-3-3437-8950
Fax: +81-3-3437-8968

PN LPS5155

How to Use this On-line Reference

Copyright © 1994, 1995, 1996 by Adobe Systems Incorporated. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written consent of the publisher. Any software referred to herein is furnished under license and may only be used or copied in accordance with the terms of such license.

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.

Any references to a "PostScript printer," a "PostScript file," or a "PostScript driver" refer to printers, files, and driver programs (respectively) which are written in or support the PostScript language. The sentences in this book that use "PostScript language" as an adjective phrase are so constructed to reinforce that the name refers to the standard language definition as set forth by Adobe Systems Incorporated.

Adobe, Acrobat, the Adobe logo, and PostScript are trademarks of Adobe Systems Incorporated or its subsidiaries and may be registered in certain jurisdictions. Apple and Macintosh are registered trademarks and AppleScript is a trademark of Apple Computer, Inc. Microsoft and MS-DOS are registered trademarks and Windows is a trademark of Microsoft Corporation. Other brand or product names are the trademarks or registered trademarks of their respective holders.

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.

Acrobat Viewer Interapplication Communication On-Line Reference Contents

How to Use this On-line Reference

IAC support description

This on-line reference has a section for each type of Interapplication communication:

- Apple events
- DDE
- OLE

Go to the appropriate IAC section by clicking its bookmark.

Each section has this structure:

1. **Description of information**. A description of syntax in the reference and anything else you need to know to use the reference for that section.
2. List of objects, if applicable. Click an entry to go to its detailed reference description.
3. **List of events, messages, or methods**. An alphabetized list with links to the detailed descriptions. Click an entry to go to its detailed reference description.
4. **Event, message, or method descriptions**. Detailed descriptions of each item.
5. **IAC-specific information**. Description of associated declarations, constants, or anything else relevant to this IAC type.

Acrobat viewer constants

The last section contains the names of menus, menu items, tools, and toolbar buttons in Acrobat viewers. It also contains other constants (such as zoom types). These constants are used in the Apple events, OLE automation methods, and DDE messages.

How to Use the Apple Events Reference

Contents

This reference contains the following sections:

1. List of Apple event objects. An alphabetized list with links to the detailed descriptions. Click an entry to go to its detailed description.

2. List of Apple events by suite. An alphabetized list with links to the detailed descriptions. Click an entry to go to its detailed description.

3. Apple Event Objects descriptions. The objects presented to the Apple event interface are: application, document, AVPageView, PDPage, PDBookmark, PDAnnot, PDTextAnnot, PDLinkAnnot, menu, and menu item. This section describes each object and lists its elements, its properties, and the methods it responds to.

4. Apple Events descriptions. The description of each Apple event includes information needed to use it from AppleScript. In addition, the descriptions of Acrobat-specific events contain information needed to use them in a programming language. AppleScript users can ignore the "Apple event ID" and "Apple event Parameters" information, while programming language users generally need all the information provided for each Apple event. Programming language users should also see Apple Event Constants. Further information on the Apple events in the Required and Core suites can be found in the *Apple Event Registry: Standard Suites*.

   Many Apple events take a parameter of type *reference*. There are many ways a reference can be constructed. For further information, see the *AppleScript Language Guide*.

5. Apple Event Constants. Values of constants needed to use Apple events from a C language program instead of from AppleScript. This lists only constants for the Acrobat-specific events and only constants that do not appear in the

Apple event chapter. See *The Apple Event Registry: Standard Suites* for a list of the constants for the required and core event suites.

The constants needed to use the Apple events from a C program are included in the header file AETypes.h, located in the IAC folder of the SDK.

Note: *The prototype for* AEObjectInit() *in the Apple-supplied file* AEObjects.h *is missing a void parameter, so* AETypes.h *contains a private, modified copy.*

Conventions

The object and event descriptions have the following conventions.

Object descriptions

The abbreviation r/o is used for properties that are read-only.

Event descriptions

All AppleScript examples use the English dialect of AppleScript syntax.

The AppleScript notation used follows that used in the *AppleScript Language Guide*:

- Keywords are shown in Roman faces, parameters in italics.
- Optional items are enclosed in square brackets.
- Lists of choices from which exactly one must be chosen are enclosed in parentheses.
- A vertical bar separates items in a list from which one must be chosen.

For example, the command defined as:

```
count [ each | every ] elementType
  [ ( in | of ) object ]
```

may be written in any of the following ways (where the element type is a menu item and the object is a menu):

```
count each menu item in menu “File”

count every menu item in menu “File”

count menu item of menu “File”
```

Each AppleScript code sample assumes that it is being executed within an appropriate tell — end tell construct, as in this example:

```
tell application “Acrobat Exchange 2.1”

  …sample code here…

end tell
```

List of Apple event objects

List of Apple events by suite

Required suite

open
print
quit
run

Core suite

close
count
delete
exists
get
hide
make
move
open
quit
save
set

Acrobat viewer suite

bring to front
clear selection
close all docs
create thumbs
delete pages
delete thumbs
DrawPageToWindow
execute
find next note
find text
get info
go backward
go forward
goto
goto next
goto previous
insert pages
is toolbutton enabled
maximize
perform
print pages
read page down
read page up
remove toolbutton

List of Apple events by suite

replace pages
scroll
select text
set info

Apple Event Objects

Apple Event Objects

Application

Description

Represents the Acrobat viewer application itself.

Elements

Element	*Can be accessed…*
Document	by name, by numeric index
Menu	by name, by numeric index
Menu item	by name
AVPageView	by the name of the document displayed, by numeric index

Properties

Property	*Class*	*r/w*	*Description*
best type	type class	[r/o]	The best descriptor type.
default type	type class	[r/o]	The default descriptor type.
class	type class	[r/o]	The class.
name	string	[r/o]	The application's name.
frontmost	boolean	[r/o]	Is this the frontmost application?
version	integer	[r/o]	The version number of the application. The two upper bytes contain the major version and the two lower bytes contain the minor version. For example, Acrobat 2.0 has a major version of 2 and a minor version of 0.
active doc	reference		The active document.
active tool	string		The currently active tool. See Toolbar button names for a list of tool names.
toolbar visibility	boolean		Whether or not the toolbar is visible.
long menu visibility	boolean		Whether or not long menus are in use.

Apple Event Objects

UI language	string	[r/o]	Identifies which language the Acrobat viewer's UI is using. This is a 3 character language codes. Language codes are: DEU – German ENU – English ESP – Spanish FRA – French ITA – Italian NLD – Dutch SVE – Swedish
open dialog at startup	boolean		Whether the "Open…" dialog is shown at startup when the Acrobat viewer is launched without a PDF file to open.
show splash at startup	boolean		Whether or not the splash screen is shown at startup.
default zoom factor	small real		The default zoom factor, in percent, used for displaying new documents. For example, a value of 100 corresponds to a zoom factor of 1.0.
default zoom type	constant		The default zoom type, which determines the zoom and fit attributes when a document is opened. Valid values are "no vary", "fit page", "fit width", "fit height", and "fit visible width".
skip warnings	boolean		Whether or not to skip warning dialogs when bookmarks, notes, and links are deleted.
PS level	integer		The PostScript level to be used when printing to a PostScript printer. May be either 1 or 2.
shrink to fit	boolean		If true, the page's contents are shrunk to fit on the page when printing.

Apple Event Objects

	case sensitivity	boolean	Whether searches made using the Find command are case sensitive.
	whole word searching	boolean	Whether searches made using the Find command look for whole words only.
	note color	'RGB'	The color of the border around newly created text annotations. It is a list of three values, each between 0 and 65535. For example, to set the note color to a deep blue from AppleScript, you might use: set the note color to {0, 0, 32768}.
	text note label	string	The text that will appear in the “title bar” of all newly created text notes.

Methods

close all docs

count

is toolbutton enabled

make

open

print

quit

remove toolbutton

run

Apple Event Objects

AVPageView

Description

Represents the view of the document in its window. Documents that aren't visible don't have AVPageViews.

Elements

Element	*Can be accessed…*
PDPage	

Properties

Property	*Class*	*r/w*	*Description*
best type	type class	[r/o]	The best descriptor type
default type	type class	[r/o]	The default descriptor type
class	type class	[r/o]	The class
name	string	[r/o]	The document's name (as shown in the window's titlebar).
zoom factor	real		The current zoom factor, in percent. For example, a value of 100 corresponds to a zoom factor of 1.0.
zoom type	constant		The zooming and content fitting algorithm current employed. Valid values are "no vary", "fit page", "fit width", "fit height", and "fit visible width".
page number	integer		the number of the current displayed page

Methods

go backward

go forward

goto

goto next

goto previous

read page down

read page up

scroll

select text

zoom

Apple Event Objects

Document

Description Represents a single open document in the Acrobat viewer.

Elements

Element	*Can be accessed…*
PDPage	by numeric index
PDBookmark	by name, by numeric index
AVPageView	No document has more than one AVPageView, so you can access it using an index of one, or use the "some" keyword from AppleScript.

Properties

Property	*Class*	*r/w*	*Description*
best type	type class	[r/o]	The best descriptor type
default type	type class	[r/o]	The default descriptor type
class	type class	[r/o]	The class
name	string	[r/o]	The document’s name (as shown in the window’s titlebar).
modified	boolean	[r/o]	Whether the document has been modified enough to warrant saving.
bounds	bounding rectangle	[r/o]	The rectangle bounding the window, specified in device space (*i.e.*, in pixels). The coordinates are specified as {left, top, right, bottom}, and the point (0,0) is in the upper left corner of the screen.
file alias	alias	[r/o]	An alias to the file where the document will be saved to if no other name is supplied. This is usually the file from which the document was read.
view mode	constant		The view mode of the document (just pages, pages and thumbs, or pages and bookmarks).

Apple Event Objects

Methods

bring to front
clear selection
close
count
create thumbs
delete
delete pages
delete thumbs
find next note
find text
get info
insert pages
maximize
print pages
replace pages
save
set info

Apple Event Objects

Menu

Description Represents a menu in the Acrobat viewer.

Elements

Element	*Can be accessed…*
Menu item	by name, by numeric index.

Properties

Property	*Class*	*r/w*	*Description*
best type	type class	[r/o]	The best descriptor type.
default type	type class	[r/o]	The default descriptor type.
class	type class	[r/o]	The class.
name	string	[r/o]	The menu's name (a language-independent name that uniquely identifies the menu). See Menu names for a list of menu names.
title	string	[r/o]	The menu's title (as shown in the menu itself). This title will be in the application's UI language.

Methods count
delete

Apple Event Objects

Menu item

Description Represents a single item in a menu in the Acrobat viewer.

Elements

Element	*Can be accessed…*
None	

Properties

Property	*Class*	*r/w*	*Description*
best type	type class	[r/o]	The best descriptor type
default type	type class	[r/o]	The default descriptor type
class	type class	[r/o]	The class
name	string	[r/o]	The menu item's name (a language-independent name that uniquely identifies the menuitem). See Menu item names for a list of menu item names.
title	string	[r/o]	The menu item's title (as shown in the menu item itself). This title will be in the application's UI language.
enabled	boolean	[r/o]	Whether the menuitem is enabled
marked	boolean	[r/o]	Whether the menuitem is checked

Methods

delete

execute

Apple Event Objects

PDAnnot

Description

Represents an annotation on a page of a document. The PDLinkAnnot and PDTextAnnot classes are two specific annotation types.

Elements

Element	*Can be accessed…*
None	

Properties

Property	*Class*	*r/w*	*Description*
best type	type class	[r/o]	The best descriptor type
default type	type class	[r/o]	The default descriptor type
class	type class	[r/o]	The class
name	string		The annotation's label (text notes only)
index	integer	[r/o]	The annotation's index within the PDPage.
subtype	string	[r/o]	The subtype of the annotation
open state	boolean		Whether the annotation is open (text notes only)
bounds	a list of small real		The boundary rectangle for the annotation, in PDF space (left, top, right, bottom).
contents	string		The textual contents of the note (text notes only)
modification date	date		The date and time the annotation was last modified
color	'RGB'		The color of the border around the annotation
destination page number	integer		The number of the page to which the link's action goes when it is performed (link annotations only)

Apple Event Objects

	destination rectangle	a list of small real	Destination rectangle (specified in user space) on the destination page for the link's action (link annotations only). Coordinates are specified in the following order: (left, top, right, bottom).
	fit type	constant	Controls how the destination rectangle is fitted to the window (link annotations only). Must be one of: Left Top Zoom Fit Page Fit Width Fit Height Fit Rect Fit BBox Fit BB Width Fit BB Height These are described in Section 6.6 of the *Portable Document Format Reference Manual* and in Technical Note #5156, Updates to the *Portable Document Format Reference Manual*.
	zoom factor	small real	The zoom factor used when fit type is Left Top Zoom; ignored otherwise. Setting this property automatically sets fit type to Left Top Zoom (link annotations only)
Methods	delete		
	perform (performs the action, if any, associated with the annotation)		

Apple Event Objects

PDBookmark

Description Represents a bookmark on a page of a document.

Elements

Element	*Can be accessed…*
None	

Properties

Property	*Class*	*r/w*	*Description*
best type	type class	[r/o]	The best descriptor type
default type	type class	[r/o]	The default descriptor type
class	type class	[r/o]	The class
name	string		The bookmark’s title
index	integer	[r/o]	The bookmark’s index within the document.
destination page number	integer		The number of the page to which the bookmark’s action goes when it is performed
destination rectangle	a list of small real		Destination rectangle (specified in user space) on the destination page for the bookmark’s action. Coordinates are specified in the following order: (left, top, right, bottom).

Apple Event Objects

	fit type	constant	Controls how the destination rectangle is fitted to the window. Must be one of: Left Top Zoom — Sets a specified zoom and a specified location on the page. Fit Page — Sets the zoom factor so that the entire page into the window. Fit Width — Sets the zoom factor so that the width of the page fits into the window. Fit Height — Sets the zoom factor so that the height of the page fits into the window. Fit Rect — Sets the zoom factor so that the specified rectangle fits into the window. Fit BBox — Sets the zoom so that the rectangle enclosing all marks on the page (known as the *bounding box*) fits into the window. Fit BB Width — Sets the zoom factor so that the width of the bounding box fits into the window. Fit BB Height — Sets the zoom factor so that the height of the bounding box fits into the window.
	zoom factor	small real	The zoom factor used when fit type is Left Top Zoom; ignored otherwise. Setting this property automatically sets fit type to Left Top Zoom
Methods	delete perform		

PDLinkAnnot

Description

Represents a link annotation on a page of a document. Can only be used as the target of a make event. All other access is via the PDAnnot class. See PDAnnot for properties and elements.

Apple Event Objects

PDPage

Description

Represents a single page in a PDF file.

Elements

Element	*Can be accessed…*
PDAnnot	by numeric index

Properties

Property	*Class*	*r/w*	*Description*
best type	type class	[r/o]	The best descriptor type
default type	type class	[r/o]	The default descriptor type
class	type class	[r/o]	The class
bounds	a list of small real		The boundary rectangle for the page {left, top, right, bottom}.
number	integer	[r/o]	The page's number
rotation	integer		The rotation angle of the page (0, 90, 180, or 270).

Methods

count

delete

DrawPageToWindow (cannot be accessed via AppleScript)

move

PDTextAnnot

Description Represents a text annotation on a page of a document. Can only be used as the target of a make event. All other access is via the PDAnnot class. See PDAnnot for properties and elements.

Apple Events

Apple Events

Required suite

open

Description Opens a file.

AppleScript Syntax open *file*

AppleScript Parameters

file
(reference) The file or files to open.

Return Value None

Apple Events

Required suite

print

Description	Prints one or more files.
AppleScript Syntax	print *file*
AppleScript Parameters	file (reference) The file or files to print.
Return Value	None

Apple Events

Required suite

quit

Description	Terminates an application. See the quit event in the core suite for a variant that accepts options.
AppleScript Syntax	quit
AppleScript Parameters	None
Return Value	None

Apple Events

Required suite

run

Description	Launches an application and invokes its standard startup procedures.
AppleScript syntax	run
AppleScript Parameters	None
Return Value	None

Apple Events

Core suite

close

Description Closes one or more documents.

AppleScript Syntax close *document* [saving *saveOption*]

AppleScript Parameters

document
: (reference) The document to close

saveOption
: A constant that specifies whether or not to save a document that has been modified before quitting. Must be one of:

  yes — Save the document.

  no — Do not save the document.

  ask — Ask the user whether or not to save the document.

  The default value is ask.

Return Value None

Related events open

Apple Events

Core suite

count

Description Counts the number of elements of a particular class in an object.

AppleScript Syntax count [each | every] *elementType* [(in | of) *object*]

AppleScript Parameters

elementType
: (class type) The class name of the elements to be counted.

object
: (reference) The object whose elements are to be counted.

Return Value An integer specifying the count.

AppleScript Example

```
count PDAnnot of document "dev_acro.pdf"
count menu item of menu "View"
```

Apple Events

Core suite

delete

Description	Deletes one or more objects.
AppleScript Syntax	delete *object*
AppleScript Parameters	object (reference) The object to delete.
Return Value	None
Related events	make exists
AppleScript Example	`delete first PDBookmark of document "test.pdf"`

Apple Events

Core suite

exists

Description Tests whether or not a specified object exists.

AppleScript Syntax

```
object exists
exists object
```

AppleScript Parameters

object
(reference) Object whose existence is to be checked.

Return Value true if the object exists, false otherwise.

AppleScript Example

```
exists second document
   second document exists
```

Apple Events

Core suite

get

Description Gets the value of a property of an object

AppleScript Syntax [get] *property* [as *className*]

AppleScript Parameters

property
: (reference) The object property or data to get.

className
: (class type) The form in which the data is to be returned.

Return Value The value of the specified property. If the specified object does not exist, no result is returned.

Related events set

AppleScript Example

```
get the name of last PDBookmark
get the index of last PDBookmark as string
```

Apple Events

Core suite

hide

Description	Hides the Acrobat viewer.
AppleScript Syntax	Hide
AppleScript Parameters	None
Return Value	None
AppleScript Example	`Hide`

Apple Events

Core suite

make

Description Creates a new object.

AppleScript Syntax make [new] *objectType* [at *location*]

AppleScript Parameters

objectType
: (class type) The class of the object to create.

location
: (reference) The location at which to insert the new object

Return Value A reference to the newly created object.

Related events delete

exists

AppleScript Example

```
set myAnnot to make PDTextAnnot at beginning
set name of myAnnot to "Werner Heisenberg"
set contents of myAnnot to "Might have been here"
```

Apple Events

Core suite

move

Description Moves a PDPage object.

AppleScript Syntax move *object* to *location*

AppleScript Parameters

object
(reference) The PDPage to move.

location
(reference) The location to which *object* is moved.

Return Value A reference to the object that was moved.

AppleScript Example

```
move PDPage 3 to before PDPage 1
```

Apple Events

Core suite

open

Description Opens a file either into a visible or a hidden window

AppleScript Syntax `open` *`file`* `[invisible` *`hiddenFlag`*`]`

AppleScript Parameters

file
: (reference) The file to open.

hiddenFlag
: (boolean) If true, the file is opened into a hidden window. If false, the file is opened into a visible window. Default is false (*i.e.*, the window is visible)

Return Value None

Related events close

Apple Events

Core suite

quit

Description Terminates the Acrobat viewer.

AppleScript Syntax quit[saving *saveOption*]

AppleScript Parameters saveOption

A constant that specifies whether to save documents that have been modified before quitting. The possible values are:

yes – save the document.

no – Do not save the document.

ask – If the file has been changed, ask the user whether or not to save the documents.

The default value is ask.

Return Value None

AppleScript Example

```
quit saving yes
```

Apple Events

Core suite

save

Description Saves a document into a file. The document window's title changes to the name of the file into which it was saved. A file cannot be saved unless it has been modified.

AppleScript Syntax save *document* to *file*

AppleScript Parameters

document
: (reference) The document to be saved.

file
: (reference) The file into which the document is to be saved.

Return Value None

AppleScript Example

```
save document "dev_acro.pdf" to file
   "Macintosh HD:backup.pdf"
```

Apple Events

Core suite

set

Description Assigns one or more values to one or more variables.

AppleScript Syntax set *location* to *value*

AppleScript Parameters

location
(reference) The location whose value is to be set.

value
(expression) The value to which *location* is to be set.

Return Value None

Related events get

AppleScript Example

```
set the name of first PDBookmark to "Chapter 1"
```

Apple Events

Acrobat viewer suite

bring to front

Description	Brings the specified document's window to the front.
AppleScript Syntax	bring to front *document*
AppleScript Parameters	document (reference) The document to bring to the front.
Return Value	None
AppleScript Example	`bring to front document "AppleEvt.pdf"`
Apple event ID	`kAEBringToFront ('bfrt')`

Apple Events

Acrobat viewer suite

clear selection

Description	Clears the document's current selection, if any.
AppleScript Syntax	clear selection *document*
AppleScript Parameters	document (reference) The document whose selection is to be cleared.
Return Value	None
Related events	select text
AppleScript Example	`clear selection document "PLUGINS.PDF"`
Apple event ID	`kAEClearSelection ('clsl')`

Apple Events

Acrobat viewer suite

close all docs

Description	Closes all documents.
AppleScript Syntax	close all docs [saving *saveOption*]
AppleScript Parameters	saveOption (constant) A constant that specifies whether to save any document that was been modified before closing it. The possible values are: yes– Save the document. no– Do not save the document. ask – If the document has been modified, ask the user whether or not to save it. The default value is ask.
Return Value	None
AppleScript Example	`close all docs`
Related events	open (required suite) open (core suite)
Apple event ID	`kAECloseAllDocs ('cldc')`

Apple Events

Acrobat viewer suite

create thumbs

Description	Creates thumbnail images for all pages in the document.
AppleScript Syntax	create thumbs *document*
AppleScript Parameters	document (reference) The document in which thumbnails are to be created.
Return Value	None
Related events	delete thumbs
AppleScript Example	`create thumbs document "roadmap.pdf"`
Apple event ID	`kAECreateThumbs (‘crtb’)`

Acrobat viewer suite

delete pages

Description
Deletes the specified pages in the document (the first page in a document is page 1).

AppleScript Syntax
delete pages *document* first *firstPage* last *lastPage*

AppleScript Parameters

document
(reference) The document containing the page to delete.

firstPage
(integer) The first page to delete.

lastPage
(integer) The last page to delete.

Return Value
None

Related events
insert pages
replace pages

AppleScript Example
```
delete pages document "AppleEvt.pdf" first 1 last 3
```

Apple event ID
`kAEDeletePages (‘dlpg’)`

Apple event Parameters
`keyAEFirstPage (‘frpg’)`
`keyAELastPage (‘lapg’)`

Apple Events

Acrobat viewer suite

delete thumbs

Description	Deletes all thumbnails from the document.
AppleScript Syntax	delete thumbs *document*
AppleScript Parameters	document (reference) The document from which thumbnails are to be deleted.
Return Value	None
Related events	create thumbs
AppleScript Example	`delete thumbs document "AppleEvt.pdf"`
Apple event ID	`kAEDeleteThumbs (‘dltb’)`

Acrobat viewer suite

DrawPageToWindow

Description

Instructs the Acrobat viewer to render into a specified window instead of its own window. Use this event if you wish to have the Acrobat viewer render into your application's window. This event must be sent to a PDPage object.

See the AEView example program in the IAC directory for an example of the use of this Apple event.

AppleScript Syntax

None

Apple event ID

`kAEDrawPageToWindow ('dwpg')`

Apple event Parameters

`keyAEXFormMatrix ('xfmm') — descriptor list containing six typeShortFloat.`

Transformation matrix from user space to the device space of the window into which drawing is occurring. Elements are in the order [a b c d e f]. See Section 3.9 of the *Portable Document Format Reference Manual* for more information on transformation matrices.

`keyAEUpdateRect ('updr') — descriptor list containing four typeShortFloat.`

Region of the page to update. List elements are in the order [left top right bottom]. If this parameter is absent, the entire page is updated.

`keyAEWindow — typeLongInteger`

WindowPtr for the window into which drawing is to occur.

`keyAEThisISAGWorld ('gwld') — typeBoolean`

(Optional) Exchange 2.1 only. Indicates whether or not the target port is a GWorld. If this parameter is absent or false, the viewer assumes the WindowPtr parameter points to a legitimate window pointer. If true, the viewer assumes WindowPtr points to a GWorldPtr. The client must correctly specify the nature of the port.

Acrobat viewer suite

execute

Description	Executes the specified menu item as if the user selected it.
AppleScript Syntax	execute *menuItem*
AppleScript Parameters	menuItem (reference) The menu item to execute. See Menu item names for a list of menu item names.
Return Value	None
AppleScript Example	`activate` `execute menu item "Open"`
Apple event ID	`kAEExecute (‘exec’)`

Apple Events

Acrobat viewer suite

find next note

Description Finds and selects the next text note in a document.

AppleScript Syntax find next note *document* [wrap around *wrapToFind*]

AppleScript Parameters

document
: (reference) The document in which to find the next text note.

wrapToFind
: (boolean) Whether or not to continue the search at the beginning of a document if a note has not been found when the end of the document is reached. If true, the search wraps around, otherwise it does not. The default value is false.

Return Value The text annotation found.

Related events find text

AppleScript Example

```
find next note document "dev_acro.pdf"
```

Apple event ID

```
kAEFindNextNote (‘fnnt’)
```

Apple Events

Acrobat viewer suite

find text

Description

Finds text in a document.

AppleScript Syntax

find text *document* string *searchString* [Case sensitive *searchCase*] [whole words *wordsOnly*] [wrap around *wrapToFind*]

AppleScript Parameters

document
: (reference) The document to be searched.

searchString
: (string) The string to be found.

searchCase
: (boolean) Whether or not searching is case-sensitive. The default value is false.

wordsOnly
: (boolean) Whether or not to search only for a whole words. The default value is false.

wrapToFind
: (boolean) Whether or not to continue the search at the beginning of a document if the specified text has not been found when the end of the document is reached. If true, the search wraps around, otherwise it does not. The default value is false.

Return Value

None

Related events

find next note

AppleScript Example

```
find text document "PLUGINS.PDF" string "Develop"
   whole words true
```

Apple event ID

`kAEFindText (‘ftxt’)`

Apple event Parameters

`keyAESearchString (‘sstr’)`

`keyAECaseSensitive (‘case’)`

`keyAEWholeWordsOnly (‘whwd’)`

`keyAEWrapAround (‘wrar’)`

Acrobat viewer suite

get info

Description Gets the value of the specified key in the document's Info dictionary.

AppleScript Syntax get info *document* key *keyName*

AppleScript Parameters

document
: (reference) The document from which to get the Info dictionary entry.

keyName
: (string) The case-sensitive Info dictionary key whose value is to be obtained. The predefined keys are: Creator, Producer, CreationDate, Author, Title, Subject, and Keywords. None of these is required in the PDF file.

Return Value A string containing the specified key's value, or an empty string if the key is not found.

AppleScript Example

```
get info document "PLUGINS.PDF" key "CreationDate"
```

Apple event ID `kAEGetInfo (‘gnfo’)`

Apple event Parameters `keyAEInfoKey (‘inky’)`

Acrobat viewer suite

go backward

Description
Goes to the previous view in the stored view history. Does nothing if the current view is the first view in the history.

AppleScript Syntax
go backward *pageView*

AppleScript Parameters
pageView
(reference) The AVPageView object in which to change the view.

Return Value
None

Related events
go forward
goto
goto next
goto previous

AppleScript Example
```
go backward first AVPageView
```

Apple event ID
`kAEGoBack ('gbck')`

Acrobat viewer suite

go forward

Description
Goes to the next view in the stored view history. Does nothing if the current view is the last view in the history.

AppleScript Syntax
go forward *pageView*

AppleScript Parameters
pageView
(reference) The AVPageView object in which to change the view.

Return Value
None

Related events
go backward
goto
goto next
goto previous

AppleScript Example
```
go forward first AVPageView
```

Apple event ID
```
kAEGoForward (‘gfwd’)
```

Apple Events

Acrobat viewer suite

goto

Description	Displays the page that has the specified page number.
AppleScript Syntax	goto *pageView* page *pageNumber*
AppleScript Parameters	pageView (reference) The AVPageView object in which to change the page number. pageNumber (integer) The page number of the page to display. The first page in a document is page 1.
Return Value	None
Related events	go backward go forward goto next goto previous
AppleScript Example	`goto first AVPageView page 2`
Apple event ID	`kAEGotoPage ('gtpg')`
Apple event Parameters	`keyAEPageNumber ('pg #')`

Acrobat viewer suite

goto next

Description

Displays the page after the one currently displayed in the AVPageView. Does nothing if the current is the last page in the document.

AppleScript Syntax

goto next *pageView*

AppleScript Parameters

pageView

(reference) The AVPageView object in which to change the page number.

Return Value

None

Related events

go backward

go forward

goto

goto previous

AppleScript Example

```
goto next first AVPageView
```

Apple event ID

`kAEGotoNextPage (‘nxpg’)`

Acrobat viewer suite

goto previous

Description

Displays the page before the one currently displayed in the AVPageView. Does nothing if the current page is the first page in the document.

AppleScript Syntax

goto previous *pageView*

AppleScript Parameters

pageView

(reference) The AVPageView object in which to change the page number.

Return Value

None

Related events

go backward

go forward

goto

goto next

AppleScript Example

goto previous first AVPageView

Apple event ID

`kAEGotoPrevPage ('pvpg')`

Acrobat viewer suite

insert pages

Description

Inserts one or more pages from one document into another.

AppleScript Syntax

insert pages *destDocument* after *afterPage* from *sourceDocument* starting with *firstPage* number of pages *numPages* [insert bookmarks *copyBookmarks*]

AppleScript Parameters

destDocument
: (reference) The document to receive the inserted page or pages.

afterPage
: (integer) The page after which the inserted pages will be placed.

sourceDocument
: (reference) The document containing the page or pages to be inserted.

firstPage
: (integer) The first page to insert.

numPages
: (integer) The number of pages to insert

copyBookmarks
: (boolean) Whether or not to copy bookmarks that point to the inserted pages. Default is true.

Return Value

None

Related events

delete pages

AppleScript Example

```
insert pages document "AppleEvt.pdf" after 2 from
   document "dev_acro.pdf" starting with 1 number
   of pages 4
```

Apple event ID

`kAEInsertPages (‘inpg’)`

Apple event Parameters

`keyAEInsertAfter (‘inaf’)`

`keyAESourceDoc (‘srdc’)`

`kAESourceStartPage (‘stpg’)`

`keyAENumPages (‘nmpg’)`

`keyAEInsertBookmarks (‘inbm’)`

Apple Events

Acrobat viewer suite

is toolbutton enabled

Description	Tests whether or not the specified button is enabled.
AppleScript Syntax	is toolbutton enabled named *buttonName*
AppleScript Parameters	buttonName (string)Button name. See Toolbar button names for a list of button names.
Return Value	true if the toolbutton is enabled, false otherwise.
Related events	remove toolbutton
AppleScript Example	`is toolbutton enabled named "AcroSrch:Query"`
Apple event ID	`kAEIsToolButtonEnabled (‘tben’)`
Apple event Parameters	`keyAEButtonname (‘tbnm’)`

Apple Events

Acrobat viewer suite

maximize

Description Sets the document's window size to be the maximum or its original size.

AppleScript Syntax maximize *document* maxsize *winSize*

AppleScript Parameters

document
: (reference) The document whose window is to be resized.

winMaximize
: (boolean) If true, the document's window is made full size. If false, the window is returned to its original size.

Return Value None

AppleScript Example

```
maximize document "AppleEvt.pdf" with max size
   false
```

Apple event ID `kAEMaximize (‘maxi’)`

Apple event Parameters `keyAEMaxSize (‘mxsz’)`

Apple Events

Acrobat viewer suite

perform

Description	Executes a bookmark's or link annotation's action.
AppleScript Syntax	perform *object*
AppleScript Parameters	object (reference) The PDBookmark or PDLinkAnnot object whose action is to be performed.
Return Value	None
AppleScript Example	`perform last PDBookmark`
Apple event ID	`kAEPerform ('prfm')`

Acrobat viewer suite

print pages

Description

Prints one or more pages from a document, without displaying a modal Print dialog box.

AppleScript Syntax

print pages *document* [first *firstPage*] [last *lastPage*] [PS Level *psLevel*] [binary output *printBinary*] [shrink to fit *printShrinkToFit*]

AppleScript Parameters

document
: (reference) The document containing the page or pages to be printed.

firstPage
: (integer) The first page to be printed (defaults to first page in the document).

lastPage
: (integer) The last page to print (defaults to last page in the document).

psLevel
: (integer) The PostScript language level (1 or 2) to use when printing to a PostScript printer. The default value is 1.

printBinary
: (boolean) Whether binary output is OK (Used for PostScript printing only). The default value is false.

printShrinkToFit
: (boolean) Whether or not pages should be shrunk to fit paper in printer. The default value is false.

Return Value

None

AppleScript Example

```
print pages document "AppleEvt.pdf" first 1 last 3
   PS Level 2 binary output true shrink to fit true
```

Apple event ID

`kAEPrintPages (‘prpg’)`

Apple event Parameters

`keyAEFirstPage (‘frpg’)`

`keyAELastPage (‘lapg’)`

`keyAEPSLevel (‘pslv’)`

`keyAEBinaryOK (‘binO’)`

`keyAEShrinkToFit (‘s2ft’)`

Apple Events

Acrobat viewer suite

read page down

Description	Scrolls forward through the document by one screen.
AppleScript Syntax	read page down *pageView*
AppleScript Parameters	pageView (reference) The AVPageView object to be scrolled.
Return Value	None
Related events	read page up scroll
AppleScript Example	`read page down first AVPageView`
Apple event ID	`kAEReadPageDown ('pgdn')`

Apple Events

Acrobat viewer suite

read page up

Description	Scrolls backward through the document by one screen.
AppleScript Syntax	read page up *pageView*
AppleScript Parameters	pageView (reference) The AVPageView object to be scrolled.
Return Value	None
Related events	read page down scroll
AppleScript Example	`read page up first AVPageView`
Apple event ID	`kAEReadPageUp ('pgup')`

Apple Events

Acrobat viewer suite

remove toolbutton

Description	Removes the specified button from the toolbar.
AppleScript Syntax	remove toolbutton named *buttonName*
AppleScript Parameters	buttonName (string) The name of the button to remove. See Toolbar button names for a list of button names.
Return Value	None
Related events	is toolbutton enabled
AppleScript Example	`remove toolbutton named "ZoomIn"`
Apple event ID	`kAERemoveToolButton (‘rmtb’)`
Apple event Parameters	`keyAEButtonname (‘tbnm’)`

Apple Events

Acrobat viewer suite

replace pages

Description

Replaces one or more pages in a document with pages from another document.

AppleScript Syntax

replace pages *destDocument* over *destPage* from *sourceDocument* starting with *firstPage* number of pages *numPages* [merge notes *copyNotes*]

AppleScript Parameters

destDocument
: (reference) The document in which pages are to be replaced.

destPage
: (integer) The first page to replace.

sourceDocument
: (reference) The document from which the replacement page or pages are obtained.

firstPage
: (integer) The first page in sourceDocument to use.

numPages
: (integer) The number of pages to replace.

copyNotes
: (boolean) Whether or not to copy notes from sourceDocument. Default is true.

Return Value

None

Related events

delete pages

insert pages

AppleScript Example

```
replace pages document "AppleEvt.pdf" over 2 from
   document "dev_acro.pdf" starting with 1 number
   of pages 4 merge notes false
```

Apple event ID

`kAEReplacePages (‘rppg’)`

Apple event Parameters

`keyAEDestStartPage (‘dtpg’)`

`keyAESourceDoc (‘srdc’)`

`keyAESourceStartPage (‘stpg’)`

`keyAENumPages (‘nmpg’)`

`keyAEMergeNotes (‘mgnt’)`

Apple Events

Acrobat viewer suite

scroll

Description

Scrolls the view of a page by the specified amount.

AppleScript Syntax

scroll *pageView* X Amount *deltaX* Y Amount *deltaY*

AppleScript Parameters

pageView
: (reference) The AVPageView object in which to scroll the view.

deltaX
: (small integer) The amount to scroll in the horizontal direction, in pixels. Positive values move the view to the right.

deltaY
: (small integer) The amount to scroll in the vertical direction, in pixels. Positive values move the view down.

Return Value

None

Related events

read page down

read page up

AppleScript Example

```
scroll first AVPageView X Amount 20 Y Amount 100
```

Apple event ID

```
kAEScroll (‘scrl’)
```

Apple event Parameters

```
keyAEXDelta (‘xdlt’)
```

```
keyAEYDelta (‘ydlt’)
```

Apple Events

Acrobat viewer suite

select text

Description

Selects the specified text.

AppleScript Syntax

select text *pageView* ([from words *wordPairs*] | [from chars *charPairs*])

AppleScript Parameters

pageView
: (reference) The AVPageView object in which to select text.

wordPairs
: (list of integer pairs) Words to select. One or more pairs of word offsets (from beginning of document) and word lengths (number of contiguous words).

charPairs
: (list of integer pairs) Characters to select. One or more pairs of character offsets (from beginning of document) and character lengths (number of contiguous characters).

Return Value

None

Related events

clear selection

AppleScript Example

```
repeat with i from 1 to 10
    repeat with j from 1 to (10 - i)
        select text from words {i, j}
    end repeat
end repeat
```

Apple event ID

`kAESetTextSelection ('stxs')`

Apple event Parameters

`keyAEWordList ('fmwd')`

`keyAECharList ('fmch')`

Apple Events

Acrobat viewer suite

set info

Description Sets the value of a specified key in the document's Info dictionary

AppleScript Syntax set info *document* key *keyName* value *newValue*

AppleScript Parameters

document
: (reference) The document in which to set the value of an Info dictionary entry.

keyName
: (string) The Info dictionary key whose value is to be set.

newValue
: (string) The value to set. All keys in the Info dictionary have strings as values.

Return Value None

AppleScript Example

```
set info document "PlugIns.pdf" key "Author"
   value "Wolfgang Pauli"
```

Apple event ID `kAESetInfo (‘snfo’)`

Apple event Parameters `keyAEInfoKey (‘inky’)`

`keyAEInfoValue (‘invl’)`

Acrobat viewer suite

zoom

Description Changes the zoom level of specified AVPageView.

AppleScript Syntax zoom *pageView* to *newZoomLevel*

AppleScript Parameters

pageView
: (reference) The AVPageView object to be zoomed.

newZoomLevel
: (real) The zoom factor, in percent (*i.e.*, a zoom factor of 100 displays the document with a magnification of 1.0).

Return Value None

AppleScript Example

```
zoom first AVPageView to 150
```

Apple event ID `kAEZoomTo (‘zmto’)`

Apple event Parameters `keyAEZoomFactor (‘zmft’)`

Apple Event Constants

Apple Event Constants

AEFixedMatrix

```
typedef struct{
  Fixed a;
  Fixed b;
  Fixed c;
  Fixed d;
  Fixed h;
  Fixed v;
} AEFixedMatrix;
```

Apple Event Constants

AEFixedPoint

```
typedef struct {
  Fixed h;
  Fixed v;
}AEFixedPoint;
```

Apple Event Constants

AEFixedRect

```
typedef struct _t_AEFixedRect {
  Fixed left;
  Fixed top;
  Fixed right;
  Fixed bottom;
} AEFixedRect;
```

Apple Event Constants

AEFloatPoint

```
typedef struct {
  float x;
  float y;
} AEFloatPoint;
```

Apple Event Constants

AEInt16Point

```
typedef struct {
  short x;
  short y;
} AEInt16Point;
```

Apple Event Constants

cApplication object properties

```
#define pActiveDoc'padc'
#define pActiveToolType'patl'
#define pToolBarIsVisible'ptbv'
#define pMenuBarIsVisible'pmbv'
#define pMenuState'pmst'
#define pLanguage'plan'
#define pOpenDialogAtStartup'pods'
#define pShowSplashAtStartup'psss'
#define pDefaultZoomFactor'pdzf'
#define pDefaultZoomType'pdzt'// an enum
#define kAEZoomNoVary'pznv'
#define kAEZoomFitPage'pzfp'
#define kAEZoomFitWidth'pzfw'
#define kAEZoomFitHeight'pzfh'
#define kAEZoomFitVisibleWidth'pzvw'
#define pSkipWarnings'pskw'
#define pPSLevel'ppsl'
#define pShrinkToFit'pstf'
#define pCaseSensitive'pcas'
#define pWholeWords'pwwd'
#define pNoteColor'pntc'
#define pNoteLabel'pntl'
```

Apple Event Constants

cAVPageView object properties

```
#define pZoomFactor'pzmf'
#define pZoomType'pzmt'// an enum
#define kAEZoomNoVary'pznv'
#define kAEZoomFitPage'pzfp'
#define kAEZoomFitWidth'pzfw'
#define kAEZoomFitHeight'pzfh'
#define kAEZoomFitVisibleWidth'pzvw'
#define pPageNumber'ppg#'
```

Apple Event Constants

cDocument object properties

```
#define pFileAlias'pfal'
#define pInstID'pIID'
#define pPermID'pPID'
#define pViewMode'pvew'// an enum
#define kAENotVisible'pnvs'
#define kAEPagesOnly'pgs '
#define kAEPagesThumbs'pgtb'
#define kAEPagesBookmarks'pgbm'
```

Apple Event Constants

cMenu and/or cMenuItem object properties

```
#define pTitle'ptit'
#define pMarked'pmrk'
```

Apple Event Constants

cPDAnnot object properties

```
#define pSubtype'psub'
#define pOpen'popn'
#define pContents'pcon'
#define pDate'pdte'
#define pIsValid'pvld'
```

Apple Event Constants

cPDBookmark object properties

```
#define pIsValid'pvld'
#define pDestPageNumber'pdp#'
#define pFitType'pftt'
#define kAEFitXYZ'pxyz'
#define kAEFitPage'pzfp'// same as kAEZoomFitPage
#define kAEFitWidth'pzfw'// same as kAEZoomFitWidth
#define kAEFitHeight'pzfh'// same as kAEZoomFitHeight
#define kAEFitRect'pfrt'
#define kAEFitBBox'pfbb'
#define kAEFitBBWidth'pfbw'
#define kAEFitBBHeight'pfbh'
#define pZoomFactor'pzmf'
#define pDestRect'pdrt'
```

Apple Event Constants

cPDPage object properties

```
#define pPageNumber'ppg#'
```

Apple Event Constants

Event class

```
#define kAEAcrobatViewerClass 'CARO'
#define kAEAcrobatViewerSuite kAEAcrobatViewerClass
```

Apple encourages the use of an application's signature as the name of its class for application-specific Apple events, hence CARO is the name of the class for Acrobat viewer-specific Apple events.

Apple Event Constants

Objects and properties

cApplication

```
typeApplicationProperty ='Papp',/* AVApp property */
```

cAVPageView

```
cAVPageView ='cpgv',/* object */
typeAVPageViewProperty ='Ppgv',/* property */
```

cDocument

```
/* AVDoc object, includes most PDDoc methods as well */
typeDocumentProperty ='Pdoc',/* AVDoc property */
```

cMenu

```
typeMenuProperty ='Pmnu',/* AVMenu property */
```

cMenuItem

```
typeMenuItemProperty ='Pmen' /* AVMenuItem property */
```

cPDAnnot

```
cPDAnnot ='cant',/* object */
typePDAnnotProperty ='Pant'/* property */
```

The following should only be used as parameters to the kAECreateElements call.

```
cPDTextAnnot ='ctan',/* PDTextAnnot object */
cPDLinkAnnot = 'clan'/* PDLinkAnnot object */
```

cPDBookmark

```
cPDBookmark ='cbkm',/* object */
typePDBookmarkProperty ='Pbkm',/* property */
```

cPDPage

```
cPDPage ='cpag',/* object */
typePDPageProperty ='Ppag',/* property */
```

How to Use the OLE On-line Reference

Contents

This reference contains the following sections:

1. List of OLE Objects. An alphabetized list with links to the detailed descriptions. Click an entry to go to its detailed description.

2. List of OLE methods by object. An alphabetized list with links to the detailed descriptions. Click the entry to go to its detailed description.

3. OLE Automation Objects descriptions. The Acrobat objects represented as OLE objects.

4. OLE Automation Methods descriptions. Detailed description of each OLE method, including its parameters, return value, and related methods.

5. OLE Declarations. Description of some data types used in OLE methods.

The header files needed by C and C++ programmers to use the OLE automation support described in this section are located in the IAC directory. of the SDK. Visual Basic users do not need these header files.

Conventions

The syntax used in this section follows that used in Microsoft Visual Basic 3.0.

List of OLE Objects

AcroExch.App
AcroExch.AVDoc
AcroExch.AVPageView
AcroExch.Hilite
AcroExch.PDAnnot
AcroExch.PDBookmark
AcroExch.PDDoc
AcroExch.PDPage
AcroExch.PDTextSelect

List of OLE methods by object

AcroExch.App

CloseAllDocs
Exit
GetActiveDoc
GetActiveTool
GetAVDoc
GetFrame
GetNumAVDocs
GetLanguage
GetPreference
Hide
Lock
Maximize
MenuItemExecute
MenuItemIsEnabled
MenuItemIsMarked
MenuItemRemove
SetActiveTool
SetFrame
SetPreference
Show
ToolButtonIsEnabled
ToolButtonRemove
Unlock

AcroExch.AVDoc

BringToFront
ClearSelection
Close
FindText
GetAVPageView
GetFrame
GetPDDoc
GetTitle
GetViewMode
IsValid
Maximize
Open
OpenInWindow
PrintPages
SetFrame
SetTextSelection
SetTitle
SetViewMode
ShowTextSelect

List of OLE methods by object

AcroExch.AVPageView

DevicePointToPage
DoGoBack
DoGoForward
GetAVDoc
GetDoc
GetPage
GetPageNum
GetZoom
GetZoomType
Goto
PointToDevice
ReadPageDown
ReadPageUp
ScrollTo
ZoomTo

AcroExch.HiliteList

Add

AcroExch.PDAnnot

GetColor
GetContents
GetDate
GetRect
GetSubtype
GetTitle
IsEqual
IsOpen
IsValid
Perform
SetColor
SetContents
SetDate
SetOpen
SetRect
SetTitle

AcroExch.PDBookmark

Destroy
GetByTitle
GetTitle
IsValid
Perform
SetTitle

List of OLE methods by object

AcroExch.PDDoc

AcquirePage
ClearFlags
Close
Create
CreateTextSelect
CreateThumbs
DeletePages
DeleteThumbs
GetFileName
GetFlags
GetInfo
GetInstanceID
GetNumPages
GetPageMode
GetPermanentID
InsertPages
MovePage
Open
OpenAVDoc
ReplacePages
Save
SetFlags
SetInfo
SetPageMode

AcroExch.PDPage

AddAnnot
AddNewAnnot
CreatePageHilite
CreateWordHilite
Draw
GetAnnot
GetAnnotIndex
GetDoc
GetNumAnnots
GetNumber
GetRotate
GetSize
RemoveAnnot
SetRotate

AcroExch.PDTextSelect

Destroy
GetBoundingRect
GetNumText
GetPage
GetText

OLE Automation Objects

OLE Automation Objects

AcroExch.App

The 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 menubar 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.

OLE Automation Objects

AcroExch.AVDoc

A document as seen in the user interface. The AVDoc object represents the window containing the open PDF file. Your application can use this object for several purposes. It can have Exchange render into the application's own window. This interface creates a window that closely resembles those that Exchange displays. It can also be used for selecting text, finding text, or printing pages. In addition, this object has several bridge methods to access other objects.

OLE Automation Objects

AcroExch.AVPageView

The window pane in which the document is drawn. The AVPageView object controls the contents of the AVDoc window, allowing your application to scroll, magnify, and go to the next , previous or arbitrary page. The history stack can be traversed through this object as well. This object, AcroExch.AVDoc, and AcroExch.App are the primary objects by which you control the user interface.

OLE Automation Objects

AcroExch.Hilite

A highlighted region of text in a PDF document. This object has a single method and is used by the PDPage object to create PDTextSelect objects.

OLE Automation Objects

AcroExch.PDAnnot

An annotation on a page in the PDF file. PDAnnots are the interface through which you can manipulate link and text annotations. Physical attributes of the annotation can be set and queried. Or the annotation can be performed when this makes sense: you can perform a link annotation but can’t perform a text annotation.

OLE Automation Objects

AcroExch.PDBookmark

A bookmark in a PDF file. The PDBookmark object has somewhat limited use through the IAC interface. If you know the title of a bookmark, you can perform it, change its title, or delete the bookmark.

OLE Automation Objects

AcroExch.PDDoc

The underlying PDF representation of a document. The PDDoc object is the hidden object behind every AVDoc. It is closest to the contents of a PDF file (at the IAC level). 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.

OLE Automation Objects

AcroExch.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. PDPages provide another opportunity for Exchange to render into your application's window. You can also access page size and rotation through the PDPage object. Text selection regions can be set up. Annotations can be created/accessed through PDPages. The first page in a PDDoc is page 0.

AcroExch.PDTextSelect

A selection of text in the Acrobat viewer, as if a user had used the mouse to select a region of text. The PDTextSelect object serves two purposes. If selected text exists within an AVDoc, your application can access the words in that region through this object. If you wish to make text appear selected in an AVDoc, you create a PDTextSelect.

OLE Automation Methods

OLE Automation Methods

AcroExch.App

CloseAllDocs

```
BOOL CloseAllDocs();
```

Description	Closes all open documents.
Parameters	None.
Return Value	Always returns true.
Related Methods	AVDoc.Close() AVDoc.Open() AVDoc.OpenInWindow() PDDoc.Close() PDDoc.Open() PDDoc.OpenAVDoc()

OLE Automation Methods

AcroExch.App

Exit

```
BOOL Exit();
```

Description	Exits the Acrobat viewer.
Parameters	None.
Return Value	Always returns true.
Related Methods	

OLE Automation Methods

AcroExch.App

GetActiveDoc

```
LPDISPATCH GetActiveDoc();
```

Description	Gets the frontmost document.
Parameters	None.
Return Value	The frontmost document.
Related Methods	App.GetAVDoc()

OLE Automation Methods

AcroExch.App

GetActiveTool

```
CString GetActiveTool();
```

Description	Gets the name of the currently active tool.
Parameters	None.
Return Value	false if the call fails. Returns the name of the currently active tool otherwise. See Tool names for a list of tool names.
Related Methods	App.SetActiveTool()

OLE Automation Methods

AcroExch.App

GetAVDoc

```
LPDISPATCH GetAVDoc(long nIndex);
```

Description

Gets an AcroExch.AVDoc by its index in the list of open AVDocs. Use App.GetNumAVDocs() to determine the number of AcroExch.AVDocs.

Parameters

nIndex

The index of the document to get.

Return Value

The specified document, or NULL if nIndex is greater than the number of open documents.

Related Methods

App.GetActiveDoc()

AcroExch.App

GetFrame

```
LPDISPATCH GetFrame();
```

Description

Gets the window's frame.

Note: GetFrame() *is not useful when the PDF file was opened with AVDoc.*OpenInWindow*(). GetFrame() returns the application window's frame—not the document window's frame. However, the application's window is hidden when a document is opened using* OpenInWindow*() and does not change in size as document windows are moved and resized.*

Parameters

None.

Return Value

The window's frame, specified as an AcroExch.Rect.

Related Methods

App.Maximize()

App.SetFrame()

OLE Automation Methods

AcroExch.App

GetNumAVDocs

```
long GetNumAVDocs();
```

Description	Gets the number of open AcroExch.AVDocs. The Acrobat viewer currently supports a maximum of 10 open documents.
Parameters	None.
Return Value	The number of open AcroExch.AVDocs.
Related Methods	App.GetActiveDoc() App.GetAVDoc()

OLE Automation Methods

AcroExch.App

GetLanguage

```
CString GetLanguage();
```

Description	Gets a code that specifies which language the Acrobat viewer's user interface is using.
Parameters	None.
Return Value	String containing a three-letter language code. Must be one of the following: DEU – German ENU – English ESP – Spanish FRA – French ITA – Italian NLD – Dutch SVE – Swedish
Related Methods	App.GetPreference() App.SetPreference()

OLE Automation Methods

AcroExch.App

GetPreference

```
long GetPreference(short nType);
```

Description

Gets a value from the preferences file. Zoom values (used in avpDefaultZoomScale and avpMaxPageCacheZoom) are returned as percentages, e.g., 1.00 is returned as 100. Colors (used in avpNoteColor) are automatically converted to RGB values from the representation used in the preferences file.

Parameters

nType

The name of the preferences item whose value is obtained.

Return Value

The value of the specified preference item.

Related Methods

App.GetLanguage()

App.SetPreference()

OLE Automation Methods

AcroExch.App

Hide

```
BOOL Hide();
```

Description	Hides the Acrobat viewer. When the viewer is hidden, the user has no control over it, and the Acrobat viewer exits when the last automation object is closed.
Parameters	None.
Return Value	Always returns true.
Related Methods	App.Show()

AcroExch.App

Lock

```
BOOL Lock(LPCSTR szLockedBy);
```

Description

Locks the Acrobat viewer. This method only needs to be called when using AVDoc.OpenInWindow() to draw into another application's window, and it must be called before invoking that method. Locking the viewer disables all other access to it — neither the user nor other programs will be able to use AcroExch until the App.Unlock() method is called. App.Unlock() must be called when you are done using OLE automation.

Parameters

szLockedBy

A string that is used as the name of the application that has locked the Acrobat viewer.

Return Value

true if the Acrobat viewer was locked successfully, false if it was not. Locking will fail if the user already has AcroExch.EXE open.

Related Methods

App.Unlock()

OLE Automation Methods

AcroExch.App

Maximize

```
BOOL Maximize(BOOL bMaximize);
```

Description Maximizes the Acrobat viewer.

Parameters bMaximize

If true, the Acrobat viewer is maximized. If false, the Acrobat viewer is returned to its normal state.

Return Value Always returns true.

Related Methods App.GetFrame()

App.SetFrame()

OLE Automation Methods

AcroExch.App

MenuItemExecute

```
BOOL MenuItemExecute(LPCSTR szMenuItemName);
```

Description

Executes the menu item whose language-independent menu item name is specified.

Parameters

szMenuItemName

The language-independent name of the menu item to execute. See Menu item names for a list of menu item names.

Return Value

Returns true if the menu item executes successfully, false otherwise.

Related Methods

App.MenuItemIsEnabled()

App.MenuItemIsMarked()

App.MenuItemRemove()

OLE Automation Methods

AcroExch.App

MenuItemIsEnabled

```
BOOL MenuItemIsEnabled(LPCSTR szMenuItemName);
```

Description

Is the specified menu item enabled?

Parameters

szMenuItemName

The language-independent name of the menu item whose enabled state is obtained. See Menu item names for a list of menu item names.

Return Value

true if the menu item is enabled, false if it is disabled or does not exist.

Related Methods

App.MenuItemExecute()

App.MenuItemIsMarked()

App.MenuItemRemove()

OLE Automation Methods

AcroExch.App

MenuItemIsMarked

```
BOOL MenuItemIsMarked(LPCSTR szMenuItemName);
```

Description

Is the specified menu item marked?

Parameters

szMenuItemName

The language-independent name of the menu item whose marked state is obtained. See Menu item names for a list of menu item names.

Return Value

true if the menu item is marked, false if it is not marked or does not exist.

Related Methods

App.MenuItemExecute()

App.MenuItemIsEnabled()

App.MenuItemRemove()

OLE Automation Methods

AcroExch.App

MenuItemRemove

```
BOOL MenuItemRemove(LPCSTR szMenuItemName);
```

Description

Removes the menu item whose language-independent menu item is specified.

Parameters

szMenuItemName

The language-independent name of the menu item to remove. See Menu item names for a list of menu item names.

Return Value

true if the menu item was removed, false if the menu item does not exist.

Related Methods

App.MenuItemExecute()

App.MenuItemIsEnabled()

App.MenuItemIsMarked()

AcroExch.App

SetActiveTool

```
BOOL SetActiveTool(LPCSTR szButtonName, BOOL bPersistent);
```

Description

Sets the active tool to the tool whose name is specified. Also sets whether or not the tool is a one-shot tool, or should remain active.

Parameters

szButtonName

The name of the tool to set as the active tool. See Tool names for a list of tool names.

bPersistent

A suggestion as to whether the tool should stay activated after it has been used, or is a one-shot tool. If true, the Acrobat viewer is requested to leave the tool active after it has been used. If false, the Acrobat viewer reverts to the previously-active tool after this tool is used once.

Return Value

true if the tool was set, false otherwise.

Related Methods

App.GetActiveTool()

App.ToolButtonIsEnabled()

App.ToolButtonRemove()

OLE Automation Methods

AcroExch.App

SetFrame

```
BOOL SetFrame(LPDISPATCH iAcroRect);
```

Description Sets the window's frame to the specified rectangle.

Parameters iAcroRect
An AcroExch.Rect specifying the window frame.

Return Value true if the frame was set, false if iAcroRect is not of type AcroExch.Rect.

Related Methods App.GetFrame()

App.Maximize()

AcroExch.App

SetPreference

```
BOOL SetPreference(short nType, long nValue);
```

Description

Sets a value in the preferences file. Zoom values (used in avpDefaultZoomScale and avpMaxPageCacheZoom) must be passed as percentages and are automatically converted to fixed point numbers, e.g., 100 is automatically converted to 1.0. Colors (used in avpNoteColor) are automatically converted from RGB values to the representation used in the preferences file.

Parameters

nType

The preferences item whose value is set. section lists the preferences items.

nValue

The value to set.

Return Value

Always returns true.

Related Methods

App.GetLanguage()

App.GetPreference()

OLE Automation Methods

AcroExch.App

Show

```
BOOL Show();
```

Description	Shows the Acrobat viewer. When the viewer is shown, the user is in control, and the Acrobat viewer does not automatically exit when the last automation object is destroyed.
Parameters	None.
Return Value	Always returns true.
Related Methods	App.Hide()

OLE Automation Methods

AcroExch.App

ToolButtonIsEnabled

```
BOOL ToolButtonIsEnabled(LPCSTR szButtonName);
```

Description

Is the specified toolbar button enabled?

Parameters

szButtonName

The name of the button whose enabled state is obtained. See Toolbar button names for a list of button names.

Return Value

true if the button is enabled, false if it is not enabled or does not exist.

Related Methods

App.GetActiveTool()

App.SetActiveTool()

App.ToolButtonRemove()

OLE Automation Methods

AcroExch.App

ToolButtonRemove

```
BOOL ToolButtonRemove(LPCSTR szButtonName);
```

Description	Removes the specified button from the toolbar.
Parameters	szButtonName The name of the button to remove. See Toolbar button names for a list of button names.
Return Value	true if the button was removed, false otherwise.
Related Methods	App.GetActiveTool() App.SetActiveTool() App.ToolButtonIsEnabled()

OLE Automation Methods

AcroExch.App

Unlock

```
BOOL Unlock();
```

Description	Unlocks the Acrobat viewer if it was previously locked.
Parameters	None.
Return Value	Always returns true.
Related Methods	App.Lock()

OLE Automation Methods

AcroExch.AVDoc

BringToFront

```
BOOL BringToFront();
```

Description	Brings the window to the front.
Parameters	None.
Return Value	Always returns true.
Related Methods	

AcroExch.AVDoc

ClearSelection

```
BOOL ClearSelection();
```

Description	Clears the current selection.
Parameters	None.
Return Value	true if the selection was cleared, false otherwise.
Related Methods	AVDoc.SetTextSelection() AVDoc.ShowTextSelect() PDDoc.CreateTextSelect() PDPage.CreatePageHilite() PDPage.CreateWordHilite() PDTextSelect.Destroy() PDTextSelect.GetBoundingRect() PDTextSelect.GetNumText() PDTextSelect.GetPage() PDTextSelect.GetText()

AcroExch.AVDoc

Close

```
BOOL Close(BOOL bNoSave);
```

Description

Closes a document.

Parameters

bNoSave

If true, the document is closed without saving it. If false and the document has been modified, the user is asked whether or not the file should be saved.

Return Value

false if an error occurred while closing the file, true otherwise.

Related Methods

App.CloseAllDocs()

AVDoc.Open()

AVDoc.OpenInWindow()

PDDoc.Close()

PDDoc.Open()

PDDoc.OpenAVDoc()

AcroExch.AVDoc

FindText

```
BOOL FindText(LPCSTR szText, BOOL bCaseSensitive, BOOL bWholeWordsOnly,
   BOOL bReset);
```

Description

Finds the specified text, scrolls so that it is visible, and highlights it.

Parameters

szText
: The text that is to be found.

bCaseSensitive
: If true, the search is case-sensitive. If false, it is case-insensitive.

bWholeWordsOnly
: If true, the search matches only whole words. If false, it matches partial words.

bReset
: If true, the search begins on the first page of the document. If false, it begins on the current page.

Return Value

true if the text was found, false if it was not.

Related Methods

AcroExch.AVDoc

GetAVPageView

```
LPDISPATCH GetAVPageView();
```

Description	Gets the AcroExch.AVPageView associated with an AcroExch.AVDoc.
Parameters	None.
Return Value	The AcroExch.AVPageView.
Related Methods	AVDoc.GetPDDoc() AVDoc.SetViewMode() AVPageView.GetAVDoc() AVPageView.GetDoc()

AcroExch.AVDoc

GetFrame

```
LPDISPATCH GetFrame();
```

Description	Gets the rectangle specifying the window's size and location.
Parameters	None.
Return Value	An AcroExch.Rect containing the frame.
Related Methods	AVDoc.SetFrame()

OLE Automation Methods

AcroExch.AVDoc

GetPDDoc

```
LPDISPATCH GetPDDoc();
```

Description	Gets the AcroExch.PDDoc associated with an AcroExch.AVDoc.
Parameters	None.
Return Value	The AcroExch.PDDoc.
Related Methods	AVDoc.GetAVPageView() AVPageView.GetAVDoc() AVPageView.GetDoc()

AcroExch.AVDoc

GetTitle

```
CString GetTitle();
```

Description	Gets the window's title.
Parameters	None.
Return Value	The window's title.
Related Methods	AVDoc.Open() AVDoc.SetTitle() PDDoc.OpenAVDoc()

OLE Automation Methods

AcroExch.AVDoc

GetViewMode

```
long GetViewMode();
```

Description	Gets the current document view mode (pages only, pages and thumbnails, or pages and bookmarks).
Parameters	None.
Return Value	The current document view mode. Will be one of the values listed in View mode.
Related Methods	AVDoc.GetAVPageView() AVDoc.SetViewMode()

OLE Automation Methods

AcroExch.AVDoc

IsValid

```
BOOL IsValid();
```

Description	Determines whether the AcroExch.AVDoc is still valid. This method only checks whether the document has been closed or deleted; it does not check the internal structure of the document.
Parameters	None.
Return Value	true if the document can still be used, false otherwise.
Related Methods	App.GetAVDoc() AVPageView.GetAVDoc()

AcroExch.AVDoc

Maximize

```
BOOL Maximize(BOOL bMaxSize);
```

Description Maximizes the window if bMaxSize is true.

Parameters bMaxSize
Indicates whether window should be maximized.

Return Value Always returns true.

Related Methods AVDoc.GetFrame()
AVDoc.SetFrame()

OLE Automation Methods

AcroExch.AVDoc

Open

```
BOOL Open(LPCSTR szFullPath, LPCSTR szTempTitle);
```

Description

Opens a file.

Parameters

szFullPath

The full pathname of the file to open.

szTempTitle

An optional title for the window in which the file is opened. If szTempTitle is NULL or the empty string (“”), it is ignored. Otherwise, szTempTitle is used as the window title.

Return Value

true if the file was opened successfully, false if it was not.

Related Methods

App.CloseAllDocs()

AVDoc.Close()

AVDoc.GetTitle()

AVDoc.OpenInWindow()

AVDoc.SetTitle()

PDDoc.Close()

PDDoc.Open()

PDDoc.OpenAVDoc()

AcroExch.AVDoc

OpenInWindow

```
BOOL OpenInWindow(LPCSTR szFullPath, short hWnd);
```

Description

Opens a PDF file and displays it in a user-specified window. A lock must have been acquired to use this method. See App.Lock()and App.Unlock().

Note: Do not set the view mode to PDFullScreen *with* AVDoc.SetViewMode()*when using* AVDoc.OpenInWindow()*: the viewer and application will hang.*

Parameters

szFullPath
The full pathname of the file to open.

hWnd
The window into which the file is displayed.

Return Value

true if the document was opened successfully, false otherwise.

Related Methods

App.CloseAllDocs()

AVDoc.Close()

AVDoc.Open()

PDDoc.Close()

PDDoc.Open()

PDDoc.OpenAVDoc()

OLE Automation Methods

AcroExch.AVDoc

PrintPages

```
BOOL PrintPages(long nFirstPage, long nLastPage, long nPSLevel,
   BOOL bBinaryOk, BOOL bShrinkToFit);
```

Description

Prints a specified range of pages without displaying any modal Print dialogs. PrintPages always uses the default printer setting in WIN.INI.

Parameters

nFirstPage
: The first page to print. The first page in a PDDoc is page 0.

nLastPage
: The last page to print. The first page in a PDDoc is page 0.

nPSLevel
: If 1, PostScript Level 1 operators are used. If 2, PostScript Level 2 operators are also used.

bBinaryOk
: If true, binary data may be included in the PostScript program. If false, all data is encoded as 7-bit ASCII.

bShrinkToFit
: If true, the page is shrunk (if necessary) to fit within the imageable area of the printed page. If false, it is not.

Return Value

false if there were any exceptions while printing, true otherwise.

Related Methods

AcroExch.AVDoc

SetFrame

```
BOOL SetFrame(LPDISPATCH iAcroRect);
```

Description	Sets the window's size and location.
Parameters	iAcroRect AcroExch.Rect specifying the window's frame.
Return Value	Always returns true.
Related Methods	AVDoc.GetFrame()

AcroExch.AVDoc

SetTextSelection

```
BOOL SetTextSelection(LPDISPATCH iAcroPDTextSelect);
```

Description

Sets the document's selection to the specified text selection. Before calling this method, use one of the following to create the text selection:

• PDDoc.CreateTextSelect() — Creates from a rectangle

• PDPage.CreatePageHilite() — Creates from a list of character offsets and counts

• PDPage.CreateWordHilite() — Creates from a list of word offsets and counts

After calling this method, use AVDoc.ShowTextSelect() to show the selection.

Parameters

iAcroPDTextSelect
The text selection to use.

Return Value

Always returns true.

Related Methods

AVDoc.ClearSelection()

AVDoc.ShowTextSelect()

PDDoc.CreateTextSelect()

PDPage.CreatePageHilite()

PDPage.CreateWordHilite()

PDTextSelect.Destroy()

PDTextSelect.GetBoundingRect()

PDTextSelect.GetNumText()

PDTextSelect.GetPage()

PDTextSelect.GetText()

AcroExch.AVDoc

SetTitle

```
BOOL SetTitle(LPCSTR szTitle);
```

Description

Sets the window's title.

Parameters

szTitle

The title to set. This method cannot be used on document windows, but only on windows created by plug-ins.

Return Value

Always returns true.

Related Methods

AVDoc.GetTitle()

AVDoc.Open()

PDDoc.OpenAVDoc()

AcroExch.AVDoc

SetViewMode

```
BOOL SetViewMode(long nType);
```

Description

Sets the mode in which the document will be viewed (pages only, pages and thumbnails, or pages and bookmarks).

Parameters

nType

The view mode to set. Must be one of the values listed in View mode.

Return Value

false if an error occurred while setting the view mode, true otherwise.

Related Methods

AVDoc.GetAVPageView()

AVDoc.GetViewMode()

AcroExch.AVDoc

ShowTextSelect

```
BOOL ShowTextSelect();
```

Description	Changes the view so that the current text selection is visible.
Parameters	None.
Return Value	Always returns true.
Related Methods	AVDoc.ClearSelection() AVDoc.SetTextSelection() PDDoc.CreateTextSelect() PDPage.CreatePageHilite() PDPage.CreateWordHilite() PDTextSelect.Destroy() PDTextSelect.GetBoundingRect() PDTextSelect.GetNumText() PDTextSelect.GetPage() PDTextSelect.GetText()

AcroExch.AVPageView

DevicePointToPage

```
LPDISPATCH DevicePointToPage(LPDISPATCH iAcroPoint);
```

Description

Converts the coordinates of a point from device space to user space.

Note: Do not use this method: it will not be available in future versions of Exchange. Furthermore, there is no way to translate between device space and machine space.

Parameters

iAcroPoint
AcroExch.Point whose coordinates are converted.

Return Value

An AcroExch.Point containing the converted coordinates.

Related Methods

AVPageView.PointToDevice()

AcroExch.AVPageView

DoGoBack

```
BOOL DoGoBack();
```

Description	Goes to the previous view on the view history stack, if any.
Parameters	None.
Return Value	Always returns true.
Related Methods	AVPageView.DoGoForward()

AcroExch.AVPageView

DoGoForward

```
BOOL DoGoForward();
```

Description	Goes to the next view on the view history stack, if any.
Parameters	None.
Return Value	Always returns true.
Related Methods	AVPageView.DoGoBack()

AcroExch.AVPageView

GetAVDoc

```
LPDISPATCH GetAVDoc();
```

Description	Gets the AcroExch.AVDoc associated with the current page.
Parameters	None.
Return Value	The AcroExch.AVDoc.
Related Methods	AVDoc.GetAVPageView() AVDoc.GetPDDoc() AVPageView.GetDoc()

AcroExch.AVPageView

GetDoc

```
LPDISPATCH GetDoc();
```

Description	Gets the AcroExch.PDDoc corresponding to the current page.
Parameters	None.
Return Value	The AcroExch.PDDoc.
Related Methods	AVDoc.GetAVPageView() AVDoc.GetPDDoc() AVPageView.GetAVDoc()

AcroExch.AVPageView

GetPage

```
LPDISPATCH GetPage();
```

Description	Gets the AcroExch.PDPage corresponding to the current page.
Parameters	None.
Return Value	The AcroExch.PDPage.
Related Methods	AVPageView.GetPageNum() PDDoc.AcquirePage() PDDoc.GetNumPages() PDPage.GetDoc() PDPage.GetNumber() PDPage.GetRotate() PDPage.GetSize() PDTextSelect.GetPage()

AcroExch.AVPageView

GetPageNum

```
long GetPageNum();
```

Description	Gets the page number of the page. The first page in a document is page zero.
Parameters	None.
Return Value	The current page's page number.
Related Methods	PDDoc.AcquirePage() PDPage.GetDoc() PDPage.GetNumber() PDDoc.GetNumPages() PDPage.GetRotate() PDPage.GetSize() AVPageView.GetPage() PDTextSelect.GetPage()

AcroExch.AVPageView

GetZoom

```
long GetZoom();
```

Description	Gets the current zoom factor, specified as a percent, *e.g.*, 100 is returned if the magnification is 1.0.
Parameters	None.
Return Value	The current zoom factor.
Related Methods	App.GetPreference() AVPageView.GetZoomType() AVPageView.ZoomTo()

AcroExch.AVPageView

GetZoomType

```
short GetZoomType();
```

Description	Gets the current zoom type.
Parameters	None.
Return Value	Zoom type. See section in Zoom strategies for a list of zoom types.
Related Methods	App.GetPreference() AVPageView.GetZoom() AVPageView.ZoomTo()

AcroExch.AVPageView

Goto

```
BOOL GoTo(long nPage);
```

Description

Goes to the specified page.

Parameters

nPage

Page number of the destination page. The first page in a PDDoc is page 0.

Return Value

true if the Acrobat viewer successfully went to the page, false otherwise.

Related Methods

AVPageView.DoGoBack()

AVPageView.DoGoForward()

AVPageView.ReadPageDown()

AVPageView.ReadPageUp()

AVPageView.ScrollTo()

AVPageView.ZoomTo()

AcroExch.AVPageView

PointToDevice

```
LPDISPATCH PointToDevice(LPDISPATCH iAcroPoint);
```

Description

Converts the coordinates of a point from user space to device space.

Do not use this method: it will not be available in future versions of Exchange. Furthermore, there is no way to translate between device space and machine space.

Parameters

iAcroPoint

AcroExch.Point whose coordinates are converted.

Return Value

An AcroExch.Point containing the converted coordinates.

Related Methods

AVPageView.DevicePointToPage()

OLE Automation Methods

AcroExch.AVPageView

ReadPageDown

```
BOOL ReadPageDown();
```

Description	Scrolls forward through the document by “one screenfull”
Parameters	None.
Return Value	Always returns true.
Related Methods	AVPageView.DoGoBack() AVPageView.DoGoForward() AVPageView.Goto() AVPageView.ReadPageUp() AVPageView.ScrollTo() AVPageView.ZoomTo()

AcroExch.AVPageView

ReadPageUp

```
BOOL ReadPageUp();
```

Description	Scrolls backward through the document by “one screenfull”
Parameters	None.
Return Value	Always returns true.
Related Methods	AVPageViewDoGoBack() AVPageViewDoGoForward() AVPageView.Goto() AVPageView.ReadPageDown() AVPageView.ScrollTo() AVPageView.ZoomTo()

AcroExch.AVPageView

ScrollTo

```
BOOL ScrollTo(short nX, short nY);
```

Description

Scrolls to the specified location on the current page.

Parameters

nX

x–coordinate of the destination.

nY

y–coordinate of the destination.

Return Value

true if the Acrobat viewer successfully scrolled to the specified location, false otherwise.

Related Methods

AVPageView.DoGoBack()

AVPageView.DoGoForward()

AVPageView.Goto()

AVPageView.ReadPageDown()

AVPageView.ReadPageUp()

AVPageView.ZoomTo()

AcroExch.AVPageView

ZoomTo

```
BOOL ZoomTo(short nType, short nScale);
```

Description

Zooms to a specified magnification.

Parameters

nType

Zoom type. See section in Zoom strategies for a list of zoom types.

nScale

The desired zoom factor, expressed as a percent, *e.g.*, 100 is a magnification of 1.0

Return Value

true if the magnification was set successfully, false otherwise.

Related Methods

AVPageView.GetZoom()

AVPageView.GetZoomType()

AVPageView.Goto()

AVPageView.ScrollTo()

OLE Automation Methods

AcroExch.HiliteList

Add

```
BOOL Add(short nOffset, short nLength);
```

Description

Adds the highlight specified by nOffset and nLength to the current highlight list. Highlight lists are used to highlight one or more contiguous groups of characters or words on a single page.

Highlight lists are used both for character- and word-based highlighting, although a single highlight list cannot contain a mixture of character and word highlights. After creating a highlight list, use PDPage.CreatePageHilite() or PDPage.CreateWordHilite() (depending on whether the highlight list contains character or word highlights) to create a text selection from the highlight list.

Parameters

nOffset

Offset of the first word or character to highlight. The first word/character on a page has an offset of zero.

nLength

The number of consecutive words or characters to highlight.

Return Value

Always returns true.

Related Methods

PDPage.CreatePageHilite()

PDPage.CreateWordHilite()

AcroExch.PDAnnot

GetColor

```
long GetColor();
```

Description	Gets an annotation’s color.
Parameters	None.
Return Value	The annotation’s color.
Related Methods	PDAnnot.SetColor()

AcroExch.PDAnnot

GetContents

```
CString GetContents();
```

Description	Gets a text annotation’s contents.
Parameters	None.
Return Value	The annotation’s contents (currently, up to 1024 characters).
Related Methods	PDAnnot.SetContents() PDAnnot.GetDate() PDAnnot.GetRect() PDAnnot.GetSubtype() PDAnnot.GetTitle()

AcroExch.PDAnnot

GetDate

```
LPDISPATCH GetDate();
```

Description	Gets an annotation’s date.
Parameters	None.
Return Value	An AcroExch.Time object containing the date.
Related Methods	PDAnnotGetContents() PDAnnot.GetRect() PDAnnot.GetSubtype() PDAnnot.GetTitle() PDAnnot.SetDate()

AcroExch.PDAnnot

GetRect

```
LPDISPATCH GetRect();
```

Description	Gets an annotation’s bounding rectangle.
Parameters	None.
Return Value	An AcroExch.Rect containing the annotation’s bounding rectangle.
Related Methods	PDAnnot.GetContents() PDAnnot.GetDate() PDAnnot.GetSubtype() PDAnnot.GetTitle() PDAnnot.SetRect()

AcroExch.PDAnnot

GetSubtype

```
CString GetSubtype();
```

Description	Gets an annotation's subtype.
Parameters	None.
Return Value	The annotation's subtype. The built-in subtypes are "Text" and "Link".
Related Methods	PDAnnot.GetContents() PDAnnot.GetDate() PDAnnot.GetRect() PDAnnot.GetTitle()

AcroExch.PDAnnot

GetTitle

```
CString GetTitle();
```

Description	Gets a text annotation’s title.
Parameters	None.
Return Value	The annotation’s title (up to 512 characters).
Related Methods	PDAnnot.GetContents() PDAnnot.GetDate() PDAnnot.GetRect() PDAnnot.GetSubtype() PDAnnot.SetTitle()

AcroExch.PDAnnot

IsEqual

```
BOOL IsEqual(LPDISPATCH PDAnnot);
```

Description

Determines whether or not an annotation is the same as the specified annotation.

Parameters

PDAnnot

The AcroExch.PDAnnot to be tested.

Return Value

true if the annotations are the same, false otherwise.

Related Methods

PDAnnot.GetContents()

PDAnnot.GetDate()

PDAnnot.GetRect()

PDAnnot.GetSubtype()

PDAnnot.GetTitle()

PDAnnot.IsOpen()

PDAnnot.IsValid()

AcroExch.PDAnnot

IsOpen

```
BOOL IsOpen();
```

Description	Tests whether or not a text annotation is open.
Parameters	None.
Return Value	true if open, false if closed.
Related Methods	PDAnnot.GetContents() PDAnnot.GetDate() PDAnnot.GetRect() PDAnnot.GetSubtype() PDAnnot.GetTitle() PDAnnot.IsEqual() PDAnnot.IsValid() PDAnnot.SetOpen()

AcroExch.PDAnnot

IsValid

```
BOOL IsValid();
```

Description	Tests whether or not an annotation is still valid. This method is intended only to test whether or not the annotation has been deleted, not whether it is a completely valid annotation object.
Parameters	None.
Return Value	true if the annotation is valid, false otherwise.
Related Methods	PDAnnot.GetContents() PDAnnot.GetDate() PDAnnot.GetRect() PDAnnot.GetSubtype() PDAnnot.GetTitle() PDAnnot.IsEqual() PDAnnot.IsOpen()

AcroExch.PDAnnot

Perform

```
BOOL Perform(LPDISPATCH iAcroAVDoc);
```

Description	Performs a link annotation's action.
Parameters	iAcroAVDoc AcroExch.AVDoc in which the annotation is located.
Return Value	true if the action was executed successfully, false otherwise.
Related Methods	PDAnnotIsValid()

AcroExch.PDAnnot

SetColor

```
BOOL SetColor(long nRGBColor);
```

Description Sets an annotation's color.

Parameters nRGBColor
The color to use for the annotation.

Return Value true if the annotation's color was set, false if the Acrobat viewer does not support editing.

Related Methods
PDAnnot.GetColor()
PDAnnot.SetContents()
PDAnnot.SetDate()
PDAnnot.SetOpen()
PDAnnot.SetRect()
PDAnnot.SetTitle()

AcroExch.PDAnnot

SetContents

```
BOOL SetContents(LPCSTR szContents);
```

Description Sets a text annotation's contents.

Parameters szContents
The contents to use for the annotation.

Return Value false if the Acrobat viewer does not support editing, true otherwise.

Related Methods
PDAnnot.GetContents()
PDAnnot.SetColor()
PDAnnot.SetDate()
PDAnnot.SetOpen()
PDAnnot.SetRect()
PDAnnot.SetTitle()

AcroExch.PDAnnot

SetDate

```
BOOL SetDate(LPDISPATCH iAcroTime);
```

Description Sets an annotation's date.

Parameters iAcroTime
Date and time to use for the annotation.

Return Value true if the date was set, false if the Acrobat viewer does not support editing.

Related Methods
PDAnnot.GetDate()
PDAnnot.SetColor()
PDAnnot.SetContents()
PDAnnot.SetOpen()
PDAnnot.SetRect()
PDAnnot.SetTitle()

OLE Automation Methods

AcroExch.PDAnnot

SetOpen

```
BOOL SetOpen(BOOL bIsOpen);
```

Description

Opens or closes a text annotation.

Parameters

bIsOpen

If true, the annotation is open. If false, the annotation is closed.

Return Value

Always returns true.

Related Methods

PDAnnot.IsOpen()

PDAnnot.SetColor()

PDAnnot.SetContents()

PDAnnot.SetDate()

PDAnnot.SetRect()

PDAnnot.SetTitle()

AcroExch.PDAnnot

SetRect

```
BOOL SetRect(LPDISPATCH iAcroRect);
```

Description Sets an annotation's bounding rectangle.

Parameters iAcroRect
Bounding rectangle (AcroExch.Rect) to set.

Return Value true if a rectangle was supplied, false otherwise.

Related Methods
PDAnnot.GetRect()
PDAnnot.SetColor()
PDAnnot.SetContents()
PDAnnot.SetDate()
PDAnnot.SetOpen()
PDAnnot.SetTitle()

AcroExch.PDAnnot

SetTitle

```
BOOL SetTitle(LPCSTR szTitle);
```

Description

Sets a text annotation's title.

Parameters

szTitle
The title to use.

Return Value

true if the title was set, false if the Acrobat viewer does not support editing.

Related Methods

PDAnnot.GetTitle()

PDAnnot.SetColor()

PDAnnot.SetContents()

PDAnnot.SetDate()

PDAnnot.SetOpen()

PDAnnot.SetRect()

AcroExch.PDBookmark

Destroy

```
BOOL Destroy();
```

Description	Destroys a bookmark. Note that it is not possible to create a bookmark with OLE.
Parameters	None.
Return Value	false if the Acrobat viewer does not support editing (making it impossible to delete the bookmark), true otherwise.
Related Methods	PDBookmark.IsValid()

AcroExch.PDBookmark

GetByTitle

```
BOOL GetByTitle(LPDISPATCH iAcroPDDoc, LPCSTR bookmarkTitle);
```

Description

Gets the bookmark that has a specified title. The AcroExch.PDBookmark object is set to the specified bookmark as a side effect of the method; it is not the method's return value. You cannot enumerate bookmark titles with this method.

Parameters

iAcroPDDoc

The document (AcroExch.PDDoc object) containing the bookmark.

bookmarkTitle

The title of the bookmark to get. The capitalization of the title must match that in the bookmark.

Return Value

true if the specified bookmark exists (the method determines this using the PDBookmark.IsValid() method), false otherwise.

Related Methods

PDBookmark.GetTitle()

PDBookmark.SetTitle()

Example

```
BOOL b;
CAcroPDBoookmark;
bookmark = new AcroPDBookmark;
COleException e;

if (!bookmark-
  >CreateDispatch("AcroExch.PDBookmark",
  &e))
  AfxMessageBox("Failed to create
  PDBookmark object.");

b = bookmark->GetByTitle(pddoc, "Name of
  bookmark");
if (b)
   bookmark->Perform();
```

AcroExch.PDBookmark

GetTitle

```
CString GetTitle();
```

Description	Gets a bookmark's title (up to 256 characters).
Parameters	None.
Return Value	The title.
Related Methods	PDBookmark.GetByTitle() PDBookmark.SetTitle()

AcroExch.PDBookmark

IsValid

```
BOOL IsValid();
```

Description	Tests whether or not the bookmark is valid. This test only ensures that the bookmark has not been deleted; it does not thoroughly check the bookmark's data structures.
Parameters	None.
Return Value	true if the bookmark is valid, false if not.
Related Methods	PDBookmark.Destroy()

AcroExch.PDBookmark

Perform

```
BOOL Perform(LPDISPATCH iAcroAVDoc);
```

Description	Performs a bookmark's action.
Parameters	iAcroAVDoc AcroExch.AVDoc in which the bookmark is located.
Return Value	true if the action was executed successfully, false otherwise.
Related Methods	PDBookmark.IsValid()

AcroExch.PDBookmark

SetTitle

```
BOOL SetTitle(LPCSTR szNewTitle);
```

Description	Sets a bookmark's title.
Parameters	szNewTitle The title to set.
Return Value	false if the Acrobat viewer does not support editing, true otherwise.
Related Methods	PDBookmark.GetByTitle() PDBookmark.GetTitle()

AcroExch.PDDoc

AcquirePage

```
LPDISPATCH AcquirePage(long nPage);
```

Description

Acquires the specified page.

Parameters

nPage

The number of the page to acquire. The first page in a PDDoc is page 0.

Return Value

The AcroExch.PDPage object for the acquired page. Returns NULL if the page could not be acquired.

Related Methods

AVPageView.GetPage()

AVPageView.GetPageNum()

PDDoc.GetNumPages()

PDPage.GetDoc()

PDPage.GetNumber()

PDPage.GetRotate()

PDPage.GetSize()

PDTextSelect.GetPage()

AcroExch.PDDoc

ClearFlags

```
BOOL ClearFlags(long nFlags);
```

Description

Clears a document's flags. The flags indicate whether the document has been modified, whether the document is a temporary document and should be deleted when closed, and the version of PDF used in the file. Note that this method can only be used to clear, not to set, the flag bits.

This method was not available prior to Acrobat Exchange 2.1.

Parameters

nFlags

Flags to be cleared. See PDDoc.GetFlags() for a description of the flags. The flags PDDocWasRepaired, PDDocNewMajorVersion, PDDocNewMinorVersion, and PDDocOldVersion are read-only and cannot be cleared.

Return Value

Always returns true.

Related Methods

PDDoc.GetFlags()

PDDoc.SetFlags()

OLE Automation Methods

AcroExch.PDDoc

Close

```
BOOL Close();
```

Description	Closes a file.
Parameters	None.
Return Value	true if the document was closed successfully, false otherwise.
Related Methods	App.CloseAllDocs() AVDoc.Close() AVDoc.Open() AVDoc.OpenInWindow() PDDoc.Open() PDDoc.OpenAVDoc()

AcroExch.PDDoc

Create

```
BOOL Create();
```

Description	Creates a new AcroExch.PDDoc.
Parameters	None.
Return Value	true if the document is created successfully, false if it is not or if the Acrobat viewer does not support editing.
Related Methods	

AcroExch.PDDoc

CreateTextSelect

```
LPDISPATCH CreateTextSelect(long nPage, LPDISPATCH iAcroRect);
```

Description

Creates a text selection from the specified rectangle on the specified page. After creating the text selection, use the AVDoc.SetTextSelection() method of to use it as the document's selection, and use AVDoc.ShowTextSelect() to show the selection.

Parameters

nPage

The page on which the selection is created. The first page in a PDDoc is page 0.

iAcroRect

The AcroExch.Rect enclosing the region to select.

Return Value

An AcroExch.PDTextSelect containing the text selection. Returns NULL if the text selection was not created successfully.

Related Methods

AVDoc.ClearSelection()

AVDoc.SetTextSelection()

AVDoc.ShowTextSelect()

PDPage.CreatePageHilite()

PDPage.CreateWordHilite()

PDTextSelect.Destroy()

PDTextSelect.GetBoundingRect()

PDTextSelect.GetNumText()

PDTextSelect.GetPage()

PDTextSelect.GetText()

AcroExch.PDDoc

CreateThumbs

```
BOOL CreateThumbs(long nFirstPage, long nLastPage);
```

Description

Creates thumbnail images for the specified page range in a document.

Parameters

nFirstPage

First page for which thumbnail images are created. The first page in a PDDoc is page 0.

nLastPage

Last page for which thumbnail images are created. The first page in a PDDoc is page 0.

Return Value

true if thumbnail images were created successfully, false if they were not or if the Acrobat viewer does not support editing.

Related Methods

PDDoc.DeleteThumbs()

OLE Automation Methods

AcroExch.PDDoc

DeletePages

```
BOOL DeletePages(long nStartPage, long nEndPage);
```

Description

Deletes pages from a file.

Parameters

nStartPage
: The first page to delete. The first page in a PDDoc is page 0.

nEndPage
: The last page to delete. The first page in a PDDoc is page 0.

Return Value

true if the pages were successfully deleted. Returns false if they were not or if the Acrobat viewer does not support editing.

Related Methods

PDDoc.AcquirePage()

PDDoc.DeletePages()

PDDoc.GetNumPages()

PDDoc.InsertPages()

PDDoc.MovePage()

PDDoc.ReplacePages()

AcroExch.PDDoc

DeleteThumbs

```
BOOL DeleteThumbs(long nStartPage, long nEndPage);
```

Description

Deletes thumbnail images from the specified pages in a document.

Parameters

nStartPage

First page whose thumbnail image is deleted. The first page in a PDDoc is page 0.

nEndPage

Last page whose thumbnail image is deleted. The first page in a PDDoc is page 0.

Return Value

true if the thumbnails were deleted, false if they were not deleted or if the Acrobat viewer does not support editing.

Related Methods

PDDoc.CreateThumbs()

AcroExch.PDDoc

GetFileName

```
CString GetFileName();
```

Description	Gets the name of the file associated with this AcroExch.PDDoc.
Parameters	None.
Return Value	The file name, which can currently contain up to 256 characters.
Related Methods	PDDoc.Save()

AcroExch.PDDoc

GetFlags

```
long GetFlags();
```

Description

Gets a document's flags. The flags indicate whether the document has been modified, whether the document is a temporary document and should be deleted when closed, and the version of PDF used in the file.

Parameters

None.

Return Value

The document's flags, containing an OR of the following:

Flag	*Description*
PDDocNeedsSave	Document has been modified and needs to be saved.
PDDocRequiresFullSave	Document cannot be saved incrementally; it must be written using PDSaveFull.
PDDocIsModified	Document has been modified slightly (such as bookmarks or text annotations have been opened or closed), but not in a way that warrants saving.
PDDocDeleteOnClose	Document is based on a temporary file that must be deleted when the document is closed or saved.
PDDocWasRepaired	Document was repaired when it was opened.
PDDocNewMajorVersion	Document's major version is newer than current.
PDDocNewMinorVersion	Document's minor version is newer than current.
PDDocOldVersion	Document's version is older than current.
PDDocSuppressErrors	Don't display errors.

Related Methods

PDDoc.ClearFlags()

PDDoc.SetFlags()

AcroExch.PDDoc

GetInfo

```
CString GetInfo(LPCSTR szInfoKey);
```

Description	Gets the value of a specified key in the document's info dictionary. A maximum of 512 bytes are returned.
Parameters	szInfoKey The key whose value is obtained.
Return Value	The string if the value was read successfully. Returns an empty string if the key does not exist or its value cannot be read.
Related Methods	PDDoc.SetInfo()

AcroExch.PDDoc

GetInstanceID

```
CString GetInstanceID();
```

Description	Gets the instance ID (the second element) from the ID array in the document's trailer.
Parameters	None.
Return Value	A string (up to 32 characters) containing the document's instance ID.
Related Methods	PDDoc.GetPermanentID()

AcroExch.PDDoc

GetNumPages

```
long GetNumPages();
```

Description	Gets the number of pages in a file.
Parameters	None.
Return Value	The number of pages, or –1 if the number of pages cannot be determined.
Related Methods	AVPageView.GetPage() AVPageView.GetPageNum() PDDoc.AcquirePage() PDPage.GetNumber() PDTextSelect.GetPage()

AcroExch.PDDoc

GetPageMode

```
long GetPageMode();
```

Description	Gets a value indicating whether the Acrobat viewer is currently displaying only pages, pages and thumbnails, or pages and bookmarks.
Parameters	None.
Return Value	The current page mode. Will be one of the values listed in section .
Related Methods	PDDoc.SetPageMode()

AcroExch.PDDoc

GetPermanentID

```
CString GetPermanentID();
```

Description	Gets the permanent ID (the first element) from the ID array in the document's trailer.
Parameters	None.
Return Value	A string (up to 32 characters) containing the document's permanent ID.
Related Methods	PDDoc.GetInstanceID()

AcroExch.PDDoc

InsertPages

```
BOOL InsertPages(long nInsertPageAfter, LPDISPATCH iPDDocSource,
   long nStartPage, long nNumPages, BOOL bBookmarks);
```

Description

Inserts pages into a file.

Parameters

nInsertPageAfter
: The page after which pages are inserted. The first page in a PDDoc is page 0.

iPDDocSource
: The AcroExch.PDDoc containing the pages to insert.

nStartPage
: The first page in iPDDocSource to insert. The first page in a PDDoc is page 0.

nNumPages
: The number of pages to insert.

bBookmarks
: If true, bookmarks are copied from the source document. If false, they are not.

Return Value

true if the pages were successfully inserted. Returns false if they were not or if the Acrobat viewer does not support editing.

Related Methods

PDDoc.AcquirePage()

PDDoc.DeletePages()

PDDoc.GetNumPages()

PDDoc.MovePage()

PDDoc.ReplacePages()

AcroExch.PDDoc

MovePage

```
BOOL MovePage(long nMoveAfterThisPage, long nPageToMove);
```

Description

Moves a page to another location within the same document.

Parameters

nMoveAfterThisPage

The page being moved is placed after this page number. The first page in a PDDoc is page 0.

nPageToMove

Page number of the page to move. The first page in a PDDoc is page 0.

Return Value

false if the Acrobat viewer does not support editing, true otherwise.

Related Methods

PDDoc.AcquirePage()

PDDoc.DeletePages()

PDDoc.GetNumPages()

PDDoc.InsertPages()

PDDoc.ReplacePages()

AcroExch.PDDoc

Open

```
BOOL Open(LPCSTR szFullPath);
```

Description

Opens a file.

Parameters

szFullPath
Full pathname of the file to open.

Return Value

true if the document was opened successfully, false otherwise.

Related Methods

App.CloseAllDocs()

AVDoc.Close()

AVDoc.Open()

AVDoc.OpenInWindow()

PDDoc.Close()

PDDoc.OpenAVDoc()

OLE Automation Methods

AcroExch.PDDoc

OpenAVDoc

```
LPDISPATCH OpenAVDoc(LPCSTR szTitle);
```

Description

Opens a window and displays the document in it.

Parameters

szTitle
the title to be used for the window. A default title is used if szTitle is NULL or the empty string (“”).

Return Value

The AcroExch.AVDoc that was opened, or NULL if the open fails.

Related Methods

App.CloseAllDocs()

AVDoc.Close()

AVDoc.GetTitle()

AVDoc.Open()

AVDoc.OpenInWindow()

AVDoc.SetTitle()

PDDoc.Close()

PDDoc.Open()

OLE Automation Methods

AcroExch.PDDoc

ReplacePages

```
BOOL ReplacePages(long nStartPage, LPDISPATCH iPDDocSource,
   long nStartSourcePage, long nNumPages, BOOL bMergeTextAnnotations);
```

Description

Replaces pages in a file with those from a specified file. No links or bookmarks are copied from iPDDocSource, but text annotations may optionally be copied.

Parameters

nStartPage
: The first page to be replaced. The first page in a PDDoc is page 0.

iPDDocSource
: A AcroExch.PDDoc containing the new copies of pages that are replaced.

nStartSourcePage
: The first page in iPDDocSource to use as a replacement page. The first page in a PDDoc is page 0.

nNumPages
: The number of pages to replace.

bMergeTextAnnotations
: If true, text annotations from iPDDocSource are copied. If false, they are not.

Return Value

true if the pages were successfully replaced. Returns false if they were not or if the Acrobat viewer does not support editing.

Related Methods

PDDoc.AcquirePage()

PDDoc.DeletePages()

PDDoc.GetNumPages()

PDDoc.InsertPages()

PDDoc.MovePage()

OLE Automation Methods

AcroExch.PDDoc

Save

```
BOOL Save(short nType, LPCSTR szFullPath);
```

Description

Saves a document.

Parameters

nType

Specifies the way in which the file should be saved. Must be one of the following:

Flag	Description
PDSaveIncremental	Write changes only, not the complete file. This will always result in a larger file, even if objects have been deleted.
PDSaveFull	Write the entire file to the filename specified by szFullPath.
PDSaveCopy	Write a copy of the file into the file specified by szFullPath, but keep using the old file. This flag can only be specified if PDSaveFull is also used.
PDSaveCollectGarbage	Remove unreferenced objects, often reducing file size. Users are encouraged to employ this flag. This flag can only be specified if PDSaveFull is also used.

szFullPath

The new pathname to the file, if any.

Return Value	true if the document was successfully saved. Returns false if it was not or if the Acrobat viewer does not support editing.
Related Methods	PDDoc.GetFileName()

AcroExch.PDDoc

SetFlags

```
BOOL SetFlags(long nFlags);
```

Description

Sets a document's flags. The flags indicate whether the document has been modified, whether the document is a temporary document and should be deleted when closed, and the version of PDF used in the file. Note that this method can only be used to set, not to clear, the flag bits.

Parameters

nFlags

Flags to be set. See PDDoc.GetFlags() for a description of the flags. The flags PDDocWasRepaired, PDDocNewMajorVersion, PDDocNewMinorVersion, and PDDocOldVersion are read-only and cannot be set.

Return Value

Always returns true.

Related Methods

PDDoc.ClearFlags()

PDDoc.GetFlags()

OLE Automation Methods

AcroExch.PDDoc

SetInfo

```
BOOL SetInfo(LPCSTR szInfoKey, LPCSTR szBuffer);
```

Description

Sets the value of a key in a document's info dictionary.

Parameters

szInfoKey
: The key whose value is set.

szBuffer
: The value to set.

Return Value

true if the value was added successfully, false if it was not or if the Acrobat viewer does not support editing.

Related Methods

PDDoc.GetInfo()

OLE Automation Methods

AcroExch.PDDoc

SetPageMode

```
BOOL SetPageMode(long nPageMode);
```

Description	Sets the Acrobat viewer to display only pages, pages and thumbnails, or pages and bookmarks.
Parameters	nPageMode The page mode to be set. Must be one of the values listed in View mode.
Return Value	Always returns true.
Related Methods	PDDoc.GetPageMode()

AcroExch.PDPage

AddAnnot

```
BOOL AddAnnot(long nIndexAddAfter, LPDISPATCH iPDAnnot);
```

Description

Adds a specified annotation at a specified location in the page's annotation array.

Parameters

nIndexAddAfter

Location in the page's annotation array to add the annotation. The first annotation on a page has an index of zero.

iPDAnnot

AcroExch.PDAnnot to add.

Return Value

false if the Acrobat viewer does not support editing, true otherwise.

Related Methods

PDPage.AddNewAnnot()

PDPage.RemoveAnnot()

AcroExch.PDPage

AddNewAnnot

```
LPDISPATCH AddNewAnnot(long nIndexAddAfter, LPCSTR szSubType,
   LPDISPATCH iAcroRect);
```

Description

Creates a new text annotation and adds it to the page.

Parameters

nIndexAddAfter

Location in the page's annotation array to add the annotation. The first annotation on a page has an index of zero.

szSubType

Subtype of the annotation to create. Must be Text.

iAcroRect

The AcroExch.Rect bounding the annotation's location on the page.

Return Value

An AcroExch.PDAnnot object, or NULL if the annotation could not be added.

Related Methods

PDPage.AddAnnot()

PDPage.RemoveAnnot()

AcroExch.PDPage

CreatePageHilite

```
LPDISPATCH CreatePageHilite(LPDISPATCH iAcroHiliteList);
```

Description

Creates a text selection from a list of character offsets and character counts on a single page. The text selection can then be set as the current selection using AVDoc.SetTextSelection(), and the view can be set to show the selection using AVDoc.ShowTextSelect().

Parameters

iAcroHiliteList

Highlight list for which a text selection is created. Use HiliteList.Add() to create a highlight list.

Return Value

The AcroExch.PDTextSelect containing the text selection, or NULL if the selection could not be created.

Related Methods

AVDoc.ClearSelection()

AVDoc.SetTextSelection()

AVDoc.ShowTextSelect()

HiliteList.Add()

PDDoc.CreateTextSelect()

PDPage.CreateWordHilite()

PDTextSelect.Destroy()

PDTextSelect.GetBoundingRect()

PDTextSelect.GetNumText()

PDTextSelect.GetPage()

PDTextSelect.GetText()

AcroExch.PDPage

CreateWordHilite

```
LPDISPATCH CreateWordHilite(LPDISPATCH iAcroHiliteList);
```

Description

Creates a text selection from a list of word offsets and word counts on a single page. The text selection can then be set as the current selection using AVDoc.SetTextSelection(), and the view can be set to show the selection using AVDoc.ShowTextSelect().

Parameters

iAcroHiliteList

Highlight list for which a text selection is created. Use HiliteList.Add() to create a highlight list.

Return Value

The AcroExch.PDTextSelect, or NULL if the selection could not be created.

Related Methods

AVDoc.ClearSelection()

AVDoc.SetTextSelection()

AVDoc.ShowTextSelect()

HiliteList.Add()

PDDoc.CreateTextSelect()

PDPage.CreatePageHilite()

PDTextSelect.Destroy()

PDTextSelect.GetBoundingRect()

PDTextSelect.GetNumText()

PDTextSelect.GetPage()

PDTextSelect.GetText()

AcroExch.PDPage

Draw

```
Draw(short window, short displayContext, short XOrigin, short YOrigin,
   short Zoom);
```

Description

Instructs the Acrobat viewer to draw into a specified window.

Parameters

window
: HWND into which the page is to be drawn.

displayContext
: HDC to use for drawing. If NULL, the HDC for window is used.

XOrigin
: *x*–coordinate of the portion of the page to draw.

YOrigin
: *y*–coordinate of the portion of the page to draw.

Zoom
: Zoom factor at which the page is to be drawn, in percent (*i.e.*, 100 corresponds to a magnification of 1.0).

Return Value

true if there were no errors in the parameters, false otherwise.

Related Methods

AcroExch.PDPage

GetAnnot

```
LPDISPATCH GetAnnot(long nIndex);
```

Description Gets the nth index annotation on the page.

Parameters nIndex

Index (in the page's annotation array) of the annotation to get. The first annotation in the array has an index of zero.

Return Value The AcroExch.PDAnnot object.

Related Methods PDPage.GetAnnotIndex()

PDPage.GetNumAnnots()

AcroExch.PDPage

GetAnnotIndex

```
long GetAnnotIndex(LPDISPATCH iPDAnnot);
```

Description

Gets the index (in the page's annotation array) of the specified annotation.

Parameters

iPDAnnot

The AcroExch.PDAnnot whose index is obtained.

Return Value

The annotation's index.

Related Methods

PDPage.GetAnnot()

PDPage.GetNumAnnots()

AcroExch.PDPage

GetDoc

```
LPDISPATCH GetDoc();
```

Description	Gets the AcroExch.PDDoc associated with the page.
Parameters	None.
Return Value	The page's AcroExch.PDDoc.
Related Methods	AVPageView.GetPage() AVPageView.GetPageNum() PDDoc.AcquirePage() PDDoc.GetNumPages() PDPage.GetNumber() PDPage.GetRotate() PDPage.GetSize() PDTextSelect.GetPage()

AcroExch.PDPage

GetNumAnnots

```
long GetNumAnnots();
```

Description	Gets the number of annotations on the page.
Parameters	None.
Return Value	The number of annotations on the page.
Related Methods	PDPage.GetAnnot() PDPage.GetAnnotIndex()

AcroExch.PDPage

GetNumber

```
long GetNumber();
```

Description	Gets the page number of the current page. The first page in a document is page zero.
Parameters	None.
Return Value	The page number of the current page. The first page in a PDDoc is page 0.
Related Methods	AVPageView.GetPage() AVPageView.GetPageNum() PDDoc.AcquirePage() PDDoc.GetNumPages() PDPage.GetDoc() PDPage.GetRotate() PDPage.GetSize() PDTextSelect.GetPage()

OLE Automation Methods

AcroExch.PDPage

GetRotate

```
short GetRotate();
```

Description	Gets the rotation value for the current page.
Parameters	None.
Return Value	Rotation value. See section in Page rotation for a list of rotation values.
Related Methods	AVPageView.GetPage() AVPageView.GetPageNum() PDDoc.AcquirePage() PDPage.GetNumber() PDPage.GetSize() PDPage.SetRotate() PDTextSelect.GetPage()

AcroExch.PDPage

GetSize

```
LPDISPATCH GetSize();
```

Description	Gets a page's width and height.
Parameters	None.
Return Value	An AcroExch.Point containing the width and height, measured in points.
Related Methods	AVPageView.GetPage() AVPageView.GetPageNum() PDDoc.AcquirePage() PDPage.GetNumber() PDPage.GetRotate() PDTextSelect.GetPage()

AcroExch.PDPage

RemoveAnnot

```
BOOL RemoveAnnot(long nIndex);
```

Description

Removes the specified annotation from the page's annotation array.

Parameters

nIndex

Index in the page's annotation array of the annotation to delete. The first annotation on a page has an index of zero.

Return Value

false if the Acrobat viewer does not support editing, true otherwise.

Related Methods

PDPage.AddAnnot (.)

PDPage.AddNewAnnot()

PDPage.GetAnnotIndex()

AcroExch.PDPage

SetRotate

```
BOOL SetRotate(short nRotate);
```

Description	Sets the rotation for the current page.
Parameters	nRotate Rotation value. See section in Page rotation for a list of rotation values.
Return Value	false if the Acrobat viewer does not support editing, true otherwise.
Related Methods	PDPage.GetRotate()

AcroExch.PDTextSelect

Destroy

```
BOOL Destroy();
```

Description	Destroys a text selection.
Parameters	None.
Return Value	Always returns true.
Related Methods	AVDoc.ClearSelection() AVDoc.SetTextSelection() AVDoc.ShowTextSelect() PDDoc.CreateTextSelect() PDPage.CreatePageHilite() PDPage.CreateWordHilite() PDTextSelect.GetBoundingRect() PDTextSelect.GetNumText() PDTextSelect.GetPage() PDTextSelect.GetText()

AcroExch.PDTextSelect

GetBoundingRect

```
LPDISPATCH GetBoundingRect();
```

Description	Gets a text selection’s bounding rectangle.
Parameters	None.
Return Value	An AcroExch.Rect corresponding to the text selection’s bounding rectangle
Related Methods	AVDoc.ClearSelection() AVDoc.SetTextSelection() AVDoc.ShowTextSelect() PDDoc.CreateTextSelect() PDPage.CreatePageHilite() PDPage.CreateWordHilite() PDTextSelect.Destroy() PDTextSelect.GetNumText() PDTextSelect.GetPage() PDTextSelect.GetText()

AcroExch.PDTextSelect

GetNumText

```
long GetNumText();
```

Description	Gets the number of text elements in a text selection. Use this method to determine how many times to call the PDTextSelect.GetText() method to obtain all of a text selection's text.
Parameters	None.
Return Value	The number of elements in the text selection.
Related Methods	AVDoc.ClearSelection() AVDoc.SetTextSelection() AVDoc.ShowTextSelect() PDDoc.CreateTextSelect() PDPage.CreatePageHilite() PDPage.CreateWordHilite() PDTextSelect.Destroy() PDTextSelect.GetBoundingRect() PDTextSelect.GetPage() PDTextSelect.GetText()

AcroExch.PDTextSelect

GetPage

```
long GetPage();
```

Description	Gets the page number on which a text selection is located.
Parameters	None.
Return Value	The text selection's page number. The first page in a PDDoc is page 0.
Related Methods	AVDoc.ClearSelection() AVDoc.SetTextSelection() AVDoc.ShowTextSelect() AVPageView.GetPage() AVPageView.GetPageNum() PDDoc.CreateTextSelect() PDDoc.GetNumPages() PDPage.CreatePageHilite() PDPage.CreateWordHilite() PDPage.GetNumber() PDTextSelect.Destroy() PDTextSelect.GetBoundingRect() PDTextSelect.GetNumText() PDTextSelect.GetText()

AcroExch.PDTextSelect

GetText

```
CString GetText(long nTextIndex);
```

Description

Gets the text from the specified element of a text selection. To obtain all text in a text selection, use PDTextSelect.GetNumText() to determine the number of elements in the text selection, then use this method in a loop to obtain each of the elements.

Parameters

nTextIndex

The element of the text selection to get.

Return Value

The text, or an empty string if nTextIndex is greater than the number of elements in the text selection.

Related Methods

AVDoc.ClearSelection()

PDPage.CreatePageHilite()

PDDoc.CreateTextSelect()

PDPage.CreateWordHilite()

PDTextSelect.Destroy()

PDTextSelect.GetBoundingRect()

PDTextSelect.GetNumText()

PDTextSelect.GetPage()

AVDocSetTextSelection()

AVDoc.ShowTextSelect()

OLE Declarations

OLE Declarations

AcroExch.Point

```
typedef struct _t_Point {
  short x;
  short y;
} Point;
```

OLE Declarations

AcroExch.Rect

```
typedef struct _t_AcroRect {
  short left;
  short top;
  short right;
  short bottom;
} Rect;
```

OLE Declarations

AcroExch.Time

```
typedef struct _t_Time {
  short year;
  short month;
  short date;
  short hour;
  short minute;
  short second;
  short millisecond;
  short day;
} Time;
```

How to Use the DDE Reference

Contents

This reference contains the following sections:

1. List of DDE messages. An alphabetized list with links to the detailed descriptions. Click the entry of interest to go to its detailed description.

1. DDE Messages descriptions. Description of each message, its arguments, return values, and related methods.

Conventions

In the DDE message descriptions, the square bracket characters [and] in DDE messages are significant, and must be included as part of the message.

List of DDE messages

AppExit
AppHide
AppShow
CloseAllDocs
DocClose
DocDeletePages
DocFind
DocGoTo
DocInsertPages
DocOpen
DocPageDown
DocPageLeft
DocPageRight
DocPageUp
DocPrint
DocReplacePages
DocSave
DocSaveAs
DocScrollTo
DocSetViewMode
DocZoomTo
FileOpen
FilePrint
FullMenus
HideToolbar
MenuitemExecute
ShortMenus
ShowToolbar

DDE Messages

DDE Messages

AppExit

```
[AppExit()]
```

Description	Exits the Acrobat viewer. AppExit is also supported in the Acrobat Reader.
Parameters	None
Return Value	true if the Acrobat viewer exited successfully, false otherwise.
Related Messages	AppHide AppShow

DDE Messages

AppHide

```
[AppHide()]
```

Description	Iconifies or hides the Acrobat viewer.
Parameters	None
Return Value	true if the Acrobat viewer was hidden successfully, false otherwise.
Related Messages	AppExit AppShow

DDE Messages

AppShow

```
[AppShow()]
```

Description	Shows the Acrobat viewer.
Parameters	None
Return Value	true if the Acrobat viewer was shown successfully, false otherwise.
Related Messages	AppExit AppHide

DDE Messages

CloseAllDocs

```
[CloseAllDocs()]
```

Description	Closes all open documents.
Parameters	None
Return Value	true if the documents were closed successfully, false otherwise.
Related Messages	DocClose DocOpen FileOpen

DocClose

```
[DocClose(char* fullPath)]
```

Description	Closes the file fullPath without saving it and without prompting the user to save the document if it has been modified.	
Parameters	fullPath	The full pathname of the file to close.
Return Value	true if the document was closed successfully. Returns false if the document does not exist or was not closed successfully.	
Related Messages	CloseAllDocs DocOpen FileOpen	

DocDeletePages

```
[DocDeletePages(char* fullPath, long fromPage, long toPage)]
```

Description

Deletes pages between fromPage and toPage in document fullPath. Requests to delete all pages in a document are not carried out, since a document must have at least one page.

Parameters

fullPath	The full pathname of the file from which pages are being deleted.
fromPage	The page number of the first page to delete.
toPage	The page number of the last page to delete.

Return Value

true if the pages were deleted successfully. Returns false if the document specified by fullPath does not exist, if the request was to delete all the document's pages, or if the pages were not deleted successfully.

Related Messages

DocInsertPages

DocReplacePages

DocFind

```
[DocFind(char* fullPath, char* string, boolean caseSensitive,
   boolean wholeWords, boolean bReset)]
```

Description

Finds a string in a specified file. This does not use the cross-document search present in version 2.0 and later of Acrobat Exchange, but performs a page-by-page search of the specified file.

Parameters

fullPath	The full pathname of the file being searched.
string	The string to find.
caseSensitive	If true, the search is case-sensitive. If false, it is not.
wholeWords	If true, the search will only match whole words. If false, the search will match partial words.
bReset	If true, the search begins on the first page of the document. If false, the search begins on the current page.

Return Value

false if the document specified by fullPath does not exist or if the text is not found. Returns true otherwise.

DDE Messages

DocGoTo

```
[DocGoTo(char* fullPath, long pageNum)]
```

Description	Goes to the page specified by pageNum.	
Parameters	fullPath	The full pathname of the file.
	pageNum	The page number of the destination page.
Return Value	false if the document specified by fullPath does not exist, true otherwise.	

DocInsertPages

```
[DocInsertPages(char* fullPath, long insertAfterPage, char* sourcePath)]
```

Description

Inserts pages from one file into another.

Parameters

fullPath — The full pathname of the file into which pages are being inserted. This file must already be open in the Acrobat viewer.

insertAfterPage — The page number in fullPath after which pages are being inserted. insertAfterPage may be a number or one of the following special strings:

PDBeforeFirstPage— Pages are inserted at the beginning of the document.

PDLastPage— Pages are inserted at the end of the document.

sourcePath — The full pathname of the file containing the new pages to insert. *All* pages from this file will be inserted into fullPath. This file does not have to be already open in the Acrobat viewer.

Return Value

true if the pages were inserted successfully. Returns false if the document does not exist or the pages were not inserted successfully.

Related Messages

DocDeletePages

DocReplacePages

DDE Messages

DocOpen

```
[DocOpen(char* fullPath)]
```

Description	Opens a document and adds it to the list of documents known to DDE, allowing it to be manipulated by other DDE messages (*see* FileOpen()).	
Parameters	fullPath	The full pathname of the file to open.
Return Value	true if the file was opened successfully, false otherwise.	
Related Messages	CloseAllDocs DocClose FileOpen	

DocPageDown

```
[DocPageDown(char* fullPath)]
```

Description	Same as the “Page Down” keyboard accelerator; scrolls forward through the document by “one screenfull.”	
Parameters	fullPath	The full pathname of the file being scrolled.
Return Value	false if the document specified by fullPath does not exist, true otherwise.	
Related Messages	DocPageLeft DocPageRight DocPageUp DocScrollTo	

DocPageLeft

```
[DocPageLeft(char* fullPath)]
```

Description	Same as “Shift left arrow” keyboard accelerator; scrolls to the left by a small amount	
Parameters	fullPath	The full pathname of the file being scrolled.
Return Value	false if the document specified by fullPath does not exist, true otherwise.	
Related Messages	DocPageDown DocPageRight DocPageUp DocScrollTo	

DocPageRight

```
[DocPageRight(char* fullPath)]
```

Description	Same as “Shift right arrow” keyboard accelerator; scrolls to the right by a small amount.	
Parameters	fullPath	The full pathname of the file being scrolled.
Return Value	false if the document specified by fullPath does not exist, true otherwise.	
Related Messages	DocPageDown DocPageLeft DocPageUp DocScrollTo	

DocPageUp

```
[DocPageUp(char* fullPath)]
```

Description	Same as the “Page Up” keyboard accelerator; scrolls backward through the document by “one screenfull.”	
Parameters	fullPath	The full pathname of the file being scrolled.
Return Value	false if the document specified by fullPath does not exist, true otherwise.	
Related Messages	DocPageDown DocPageLeft DocPageRight DocScrollTo	

DocPrint

```
[DocPrint(char* fullPath, long startPage, long endPage)]
```

Description

Prints a specified range of pages from a document, without displaying any modal Print dialog to the user. For PostScript printing, only Level 1 operators are used, only ASCII data is generated, and the document's pages are not shrunk to fit onto the imageable area of the printed page.

Parameters

fullPath	The full pathname of the file being printed.
startPage	The page number of the first page to print.
endPage	The page number of the last page to print.

Return Value

false if the document specified by fullPath does not exist, true otherwise.

Related Messages

FilePrint

DocReplacePages

```
[DocReplacePages(char* fullPath, long startDestPage, char* sourcePath,
   long startSourcePage, long endSourcePage)]
```

Description

Replaces pages in fullPath using pages from sourcePath. fromPage is the first page in fullPath that is replaced. Pages in fullPath are replaced with pages fromSourcePage to toSourcePage from sourcePage.

Parameters

fullPath	The full pathname of the file in which pages are being replaced. This file must already be open in the Acrobat viewer.
startDestPage	The page number of the first page in fullPath that is to be replaced.
sourcePath	The full pathname of the file from which replacement pages are obtained. This file does not have to be already open in the Acrobat viewer.
startSourcePage	The page number of the first page in sourcePath to use as a replacement page.
endSourcePage	The page number of the last page in sourcePath to use as a replacement page.

Return Value

true if the pages were replaced successfully. Returns false if the document does not exist or the pages were not replaced successfully.

Related Messages

DocDeletePages

DocInsertPages

DocSave

```
[DocSave(char* fullPath)]
```

Description	Saves the specified file. The user is not warned if there are any problems saving the file.	
Parameters	fullPath	The full pathname of the file to save.
Return Value	true if the document was saved successfully. Returns false if the document does not exist or was not saved successfully.	
Related Messages	DocSaveAs	

DocSaveAs

```
[DocSaveAs(char* fullPath, char* newPath)]
```

Description	Saves an open file into a new file. The user is not warned if there are any problems saving the file.	
Parameters	fullPath	The full pathname of the existing file.
	newPath	The full pathname of the new file.
Return Value	true if the document was saved successfully. Returns false if the document does not exist or was not saved successfully.	
Related Messages	DocSave	

DocScrollTo

```
[DocScrollTo(char* fullPath, int x, int y)]
```

Description	Scrolls the view of the current page to a specified location.	
Parameters	fullPath	The full pathname of the file being scrolled.
	x	The destination's *x*–coordinate.
	y	The destination's *y*–coordinate.
Return Value	false if the document specified by fullPath does not exist, true otherwise.	
Related Messages	DocPageDown DocPageLeft DocPageRight DocPageUp	

DocSetViewMode

```
[DocSetViewMode(char* fullPath, char* viewType)]
```

Description

Controls whether bookmarks, thumbnail images, or neither are shown in addition to the document.

Parameters

fullPath	The full pathname of the file whose view mode is being set.
viewType	The view mode to use. Must be one of the following: PDUseThumbs—Displays pages and thumbnail images PDUseNone— Displays only pages PDUseBookmarks—Displays pages and bookmarks

Return Value

true if the view mode was set successfully. Returns false if the document specified by fullPath does not exist or an unknown view mode is specified.

Related Messages

FullMenus

ShortMenus

DocZoomTo

```
[DocZoomTo(char* fullPath, char* zoomType, int scale)]
```

Description

Sets the zoom for a specified document.

Parameters

fullPath — The full pathname of the file whose zoom is being set.

zoomType — The zoom strategy to use. Must be one of the following:

AVZoomNoVary — A fixed zoom, such as 100%

AVZoomFitPage — Fits the page in the window

AVZoomFitWidth — Fits the page's width into the window

AVZoomFitVisibleWidth — Fits the page's visible content into the window

scale — The magnification in percent (*i.e.*, 100 corresponds to a magnification of 1.0). scale is used only when zoomType is AVZoomNoVary.

Return Value

false if the document specified by fullPath does not exist, or if zoomType has an unknown value. Returns true otherwise.

DDE Messages

FileOpen

```
[FileOpen(char* fullPath)]
```

Description	Opens and displays a file. If the file is already open, makes it the current document and brings it to the front. This DDE message does not add the document to the list that can be manipulated using DDE message; use DocOpen() to do that. FileOpen is also supported in the Acrobat Reader.	
Parameters	fullPath	The full pathname of the file to open.
Return Value	true if the file was opened successfully, false otherwise.	
Related Messages	CloseAllDocs DocClose DocOpen	

FilePrint

```
[FilePrint(char* fullPath)]
```

Description	Prints all pages in a document, without displaying any modal Print dialog to the user. For PostScript printing, only Level 1 operators are used, only ASCII data is generated, and the document's pages are not shrunk to fit onto the imageable area of the printed page. FilePrint is also supported in the Acrobat Reader.	
Parameters	fullPath	The full pathname of the file being printed.
Return Value	false if the document specified by fullPath does not exist, true otherwise.	
Related Messages	DocPrint	

FullMenus

```
[FullMenus()]
```

Description	Displays full menus, and sets this option in the Acrobat viewer's preferences file.
Parameters	None
Return Value	true if full menus were set successfully, false otherwise.
Related Messages	DocSetViewMode ShortMenus

HideToolbar

```
[HideToolbar()]
```

Description	Hides the toolbar.
Parameters	None
Return Value	true if the toolbar was hidden successfully, false otherwise.
Related Messages	ShowToolbar

MenuitemExecute

```
[MenuitemExecute(char* menuItemName)]
```

Description	Invokes a menu item, given its language-independent name.	
Parameters	menuItemName	The language-independent name of the menu item to execute. See Menu item names for a list of menu item names.
Return Value	true if the menu item was executed successfully. Returns false if the menu item does not exist, is not enabled, or was not executed successfully.	

ShortMenus

```
[ShortMenus()]
```

Description	Displays short menus, and sets this option in the Acrobat viewer's preferences file.
Parameters	None
Return Value	true if short menus were set successfully, false otherwise.
Related Messages	DocSetViewMode FullMenus

DDE Messages

ShowToolbar

```
[ShowToolbar()]
```

Description	Shows the toolbar.
Parameters	None
Return Value	true if the toolbar was shown successfully, false otherwise.
Related Messages	HideToolbar

Acrobat Viewer Constants

Acrobat Viewer Constants

Menu names

Table 1 Menu names

Menu name	*Description*
Window	Window menu.
Edit	Edit menu.
Tools	Tools menu.
Apple	Apple menu *(Macintosh only)*
View	View menu.
Extensions	Plug-ins menu.
Pages	Pages submenu of Edit menu.
Bookmarks	Bookmarks submenu of Edit menu.
Thumbnails	Thumbnails submenu of Edit menu.
Notes	Notes submenu of Edit menu.
DocInfo	Document Info submenu of File menu.
AboutExtensions	About Acrobat Plug-ins submenu.
File	File menu.
Prefs	Preferences submenu of Edit menu.
Help	Help menu *(Windows only)*

Acrobat Viewer Constants

Menu item names

Table 2 Apple menu items (Macintosh)

About
AboutExtensions
(unnamed menu item)

Table 3 Help menu item names (Windows)

About
AboutExtensions
UsingViewer
UsingExtensions

Table 4 File menu item names

Open
Close
endFileAccessGroup
Save
SaveAs
endSaveGroup
DocInfo
GeneralInfo
FontsInfo
OpenInfo
SecurityInfo
endDocInfoGroup
PageSetup
Print
endPrintGroup
Recent File (Windows only. Note the space in the name)
endRecentFileGroup (Windows only)
Quit

Table 5 Edit menu item names

Undo
endUndoGroup
Cut
Copy
Paste
Clear
SelectAll
endEditGroup
CopyFileToClipboard (Windows only)
endOleGroup (Windows only)
Pages
CropPages
RotatePages
endPageOpGroup
InsertPages
ExtractPages
ReplacePages
DeletePages
Bookmarks
NewBookmark
SetBookmarkDest
Thumbs
CreateAllThumbs
DeleteAllThumbs
Notes
ImportNotes
ExportNotes
endObjectsEditGroup
Properties
endPropertiesGroup

ShortLongMenus
Prefs
GeneralPrefs
NotesPrefs
FullScreenPrefs

Table 6 View menu item names

ActualSize
FitPage
FitWidth
FitVisible
FullScreen
endZoomTypeGroup
ZoomTo
endZoomGroup
FirstPage
PrevPage
NextPage
LastPage
GoToPage
endPageNavGroup
ArticleThreads
endArticlesGroup
GoBack
GoForward
endGoBackGroup
PageOnly
ShowBookmarks
ShowThumbs

Acrobat Viewer Constants

Table 7 Tool menu item names

Hand
ZoomIn
ZoomOut
SelectText
SelectGraphics
Note
Link
Thread
endToolsGroup
Find
FindAgain
endFindGroup
FindNextNote
CreateNotesFile

Table 8 Window menu item names

ShowHideToolBar
ShowHideClipboard
endShowHideGroup
Cascade
TileHorizontal
TileVertical
CloseAll

Acrobat Viewer Constants

Toolbar button names

Table 9 Toolbar button names

Button name	*Description*
UseNone	Displays document only (no bookmarks or thumbnail images).
UseBookmarks	Displays document and bookmarks.
UseThumbs	Displays document and thumbnail images.
endPageModeGroup	Separator not visible in the toolbar.
Hand	Allows user to scroll the page.
ZoomIn	Zooms in.
ZoomOut	Zooms out.
Select	Allows user to select text.
Note	Create/select/edit notes.
Link	Create/select/edit links.
endToolsGroup	Separator not visible in the toolbar.
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 not visible in the toolbar.
GoBack	Goes to the previous view in the view history.
GoForward	Goes to the next view in the view history.
endPageStackGroup	Separator not visible in the toolbar.
Zoom100	Sets the zoom factor to 100% (*i.e.*, 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 on which drawing appears into the window.
endZoomGroup	Separator not visible in the toolbar.
FindDialog	Brings up the Find dialog (not the cross-document search dialog).
endDialogGroup	Separator not visible in the toolbar.

Acrobat Viewer Constants

Tool names

Table 10 Tool names

Tool name	*Description*
Hand	Hand tool.
Note	Tool for making notes.
Select	Text selection tool.
Zoom	Tool for changing the zoom factor.
Link	Link creation tool.
Thread	Thread creation tool.

Zoom strategies

Table 11 Zoom strategies

Zoom type	*Description*
AVZoomNoVary	No variable zoom (*i.e.*, zoom is a fixed value such as 100%).
AVZoomFitPage	Fit page to window.
AVZoomFitWidth	Fit page width to window.
AVZoomFitHeight	Fit page height to window.

Acrobat Viewer Constants

Page rotation

Table 12 Page rotation

Rotation
pdRotate0
pdRotate90
pdRotate180
pdRotate270

Acrobat Viewer Constants

View mode

Table 13 View mode

View mode	*Description*
PDDontCare	Leave the view mode as it is.
PDUseNone	Display the document, but neither bookmarks nor thumbnail images.
PDUseThumbs	Display the document and thumbnail images.
PDUseBookmarks	Display the document and bookmarks.
PDFullScreen	Display the document in full screen mode.

Acrobat Viewer Constants

Preferences item names

Table 14 Preferences file item names

avpCaseSensitive boolean If true, the Acrobat viewer's Find command (not the Search plug-in) performs case-sensitive searches. If false, searches are not case-sensitive.
avpDefaultOverviewType Int32 Whether thumbnail images, bookmarks, or neither should be shown along with documents by default. Must be one of the following: PDUseNone — Document only. PDUseThumbs — Document plus thumbs. PDUseBookmarks — Document plus bookmarks. PDFullScreen — Full screen mode.
avpDefaultSplitterPos Int32 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.
avpDefaultZoomScale Fixed Default magnification when a document is opened.
avpDefaultZoomType AVZoomType Default zoom type for a page view. Must be one of the values listed in Zoom strategies.
avpDestFitType char * Default destination fit type for creating links and bookmarks.
avpDestZoomInherit boolean If true, the default for creating new links and bookmarks is "Inherit zoom"
avpDoCalibratedColor boolean If true, the Acrobat viewer renders using calibrated color.

Acrobat Viewer Constants

avpEnablePageCache

boolean

If true, the following page is rendered offscreen while the current page is viewed, improving performance when a document is viewed sequentially. If false, no draw-ahead is used.

avpFullScreenChangeTimeDelay

Int32

Time (in seconds) to show each page when using automatic page changing in full screen mode.

avpFullScreencolor

PDColorValue

The background color to use when the Acrobat viewer is in Full Screen mode.

avpFullScreenLoop

boolean

If true, the document's pages are displayed in a loop (rather than just once) when using full screen mode. If false, they are not.

avpGreekLevel

Int32

Size, in points, below which text is greeked if avpGreekText is true.

avpGreekText

boolean

If true, text smaller than avpGreekLevel is greeked (displayed as gray boxes). If false, all text is drawn, regardless of its size.

avpHighlightMode

Int32 (Used on Macintosh only)

Specifies the way in which highlighted text is to be displayed. Can have one of the following values:

#define HIGHLIGHT_PAINT_XOR 0 — Paint highlight color in XOR mode

#define HIGHLIGHT_INVERT_MAC 1 — Invert in special Macintosh highlight mode

#define HIGHLIGHT_INVERT_XOR 2 —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.

avpMaxPageCacheBytes

Int32

The maximum number of bytes the page cache is allowed to occupy.

Acrobat Viewer Constants

avpMinPageCacheTicks

Int32

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.

avpMaxPageCacheZoom

Fixed

Maximum zoom factor at which pages will be cached. Pages viewed at a higher zoom factor will not cached. A value of 1.0 corresponds to a zoom factor of 100%.

avpMaxThreadZoom

Fixed

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%.

avpNoteColor

PDColorValue

Default color to use for new notes.

avpNoteLabel

char *

Default label to use for new notes.

avpOpenDialogAtStartup

boolean

If true, the “file open” dialog is displayed when the Acrobat viewer is launched without a document to open. If false, it does not.

avpPrefsVersion

Int32

(Read only) The preferences file format version number.

avpPSLevel

Int32

The PostScript language level to use when printing to a PostScript printer. Allowed values are 1 and 2.

avpRememberDialogs

boolean

If true, the Acrobat viewer remembers the location of certain dialogs (such as the Find dialog) and displays them in their previous location. If false, they are displayed in a default location.

Acrobat Viewer Constants

avpShortMenus

boolean

If true, the Acrobat viewer displays short menus. If false, it displays long menus.

avpShowLargeImages

boolean

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.

avpShowSplashAtStartup

boolean

If true, the Acrobat viewer splash screen is shown when the Acrobat viewer is launched. If false, it is not.

avpShowToolBar

boolean

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

boolean

If true, pages are shrunk to fit the imageable area of the printer when printed. If false, pages are not shrunk to fit.

avpSkipWarnings

boolean

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

Int32

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. The allowed values are:

0 — Use both sans serif and serif

1 — Use sans serif only

2— Use serif only

Acrobat Viewer Constants

avpThumbViewScale

Fixed

The scale at which thumbnail images are created. The Acrobat viewer's default is fixedEighth, creating thumbnail images whose linear dimensions are one-eighth those of the actual page.

avpWholeWords

boolean

If true, the Acrobat viewer's Find command (not the Search plug-in) matches only whole words. If false, the partial words are also matched.

Changes Since Earlier Versions

Changes since 2 April 1996 version

Fixed up tables in Acrobat Viewer Constants section. Added note to description of OLE automation method App.GetFrame().

Changes since 13 February 1996 version

- Clarified page numbering conventions in PDDocs for various platforms. Made minor corrections.

Changes since 31 January 1996 version

- Improve bookmark layout.

Changes since 15 September 1995 version

- Correct minor errors.
- Removed references to Exchange LE 2.0.