PDF-XChange Viewer ActiveX SDK

Tracker Software Products (Canada) Ltd
PO Box 79, 9622 Chemainus Rd
Chemainus. BC
V0R 1K0
Canada

http://www.tracker-software.com
Mailto:Sales@tracker-software.com

Table of Contents

Foreword 0

Part I Important - Please Read 1

1 Redistribution Information ... 3

2 ActiveX Control Installation ... 3

3 Licensing ... 5

License Text ... 5

4 Requesting your V2.x Upgrade license ... 12

Part II Reference 13

1 Interfaces ... 13

IPDFXCview ... 13

General Methods ... 14

ApplyAllCachedChanges ... 14

DiscardAllCachedChanges ... 15

DoVerb ... 15

GetProperty ... 16

GetVersionInfo ... 17

LoadSettings ... 17

RunJavaScript ... 18

SaveSettings ... 18

SetDevInfo ... 19

SetProperty ... 20

Document Methods ... 20

ActivateDocument ... 21

CloseAllDocuments ... 21

CloseDocument ... 22

DoDocumentVerb ... 23

ExportDocument ... 24

FlushDocument ... 24

GetActiveDocument ... 25

GetDocumentID ... 25

GetDocumentIndex ... 26

GetDocumentProperty ... 26

GetDocumentsCount ... 27

OpenDocument ... 28

PrintDocument ... 28

SaveDocument ... 29

SetDocumentProperty ... 30

Auxiliary Methods ... 31

GetDocumentFromName ... 31

GetTextFromResult ... 32

GetView ObjectFromName ... 32

Properties ... 33

Property ... 33

IPDFXCview2 ... 34

Properties ... 34

Allow Accelerators ... 34

LockedView ... 35

ReadOnly ... 35

SettingsURL ... 36

Src ... 36

_IPDFXCviewEvents ... 37
OnEvent ... 37
IPDFXCargs ... 38
Methods ... 40
Add ... 40
Clear ... 40
Init ... 40
Remove ... 41
Properties ... 41
_NewEnum ... 41
Count ... 42
Data ... 42
Item ... 42
IPDFXCsmartp ... 43
Methods ... 44
DoVerb ... 44
GetItemPointByID ... 45
GetItemPointByIndex ... 45
GetItemPointByNIndex ... 46
GetPointName ... 46
GetProperty ... 47
GetSubPoint ... 47
SetProperty ... 47
Properties ... 47
Property ... 48
2 Named Items ... 48
Simple Operations ... 48
.ARGS ... 50
.SP ... 50
ActivateDocument ... 51
ApplyAllCachedChanges ... 51
ClearRecentsList ... 52
CloseAllDocuments ... 52
CloseDocument ... 52
CreateNewBlankDocument ... 53
DeleteDocumentPages ... 53
DiscardAllCachedChanges ... 54
ExecuteCommand ... 54
ExportDocument ... 54
ExtractDocumentPages ... 55
FlushDocument ... 55
GetActiveDocument ... 56
GetDocumentFromName ... 56
GetDocumentID ... 57
GetDocumentIndex ... 57
GetDocumentsCount ... 57
GetTextFromResult ... 58
GetViewObjectFromName ... 58
HasVisibleBars ... 59
InsertDocumentPages ... 59
InsertEmptyDocumentPages ... 60
IsPDF ... 60
LoadSettings ... 61
MsgToCommand ... 61
MsgToCommandAndExec ... 62
NewBlankDocument ... 63
NewDocumentFromImages ... 63
NewDocumentFromRTF ... 63
NewDocumentFromText ... 64

OpenDocument ... 65
PrintDocument ... 66
RemoveRecentItem ... 66
RotateDocumentPages ... 67
RunJavaScript ... 67
SaveDocument ... 68
SaveSettings ... 69
Show StampsCollection ... 69
SummarizeDocumentAnnots ... 69
Objects ... 70
ColorManagement ... 72
<Item> ... 73
Commands ... 74
Command List ... 74
<Item> ... 97
Shortcut ... 97
Commenting ... 98
Area ... 100
Styles ... 100
<Item> ... 100
Arrow ... 101
Styles ... 101
<Item> ... 102
Callout ... 103
Styles ... 103
<Item> ... 104
Cloud ... 104
Styles ... 105
<Item> ... 105
Distance ... 106
Styles ... 106
<Item> ... 107
FileAttachment ... 108
Styles ... 108
<Item> ... 109
Highlight ... 109
Styles ... 110
<Item> ... 110
Line ... 111
Styles ... 111
<Item> ... 112
Link ... 113
Styles ... 113
<Item> ... 113
Oval ... 114
Styles ... 114
<Item> ... 115
Pencil ... 116
Styles ... 116
<Item> ... 116
Perimeter ... 117
Styles ... 117
<Item> ... 118
Polygon ... 119
Styles ... 119
<Item> ... 120
Polyline ... 120
Styles ... 120
<Item> ... 121

Rect 122
Styles 122
<Item> 123
Stamp 123
Styles 124
<Item> 124
StickyNote 125
Styles 125
<Item> 126
StrikeOut 126
Styles 127
<Item> 127
TextBox 128
Styles 128
<Item> 129
Typewriter 129
Styles 130
<Item> 130
Underline 131
Styles 131
<Item> 132
Documents 133
<Item> 135
<Methods> 138
AddAttachment 138
ClearOperationsHistory 138
ClearSelection 139
Close 139
DeleteAllAttachments 140
DeleteAttachment 140
DeletePages 141
Export 141
ExtractPages 142
Flush 142
GetAllSelectedText 143
GetAllText 143
GetAttachmentDesc 144
GetAttachmentModDate 145
GetAttachmentName 145
GetAttachmentsCount 146
GetAttachmentSize 146
GetSelectedAnnot 147
GetSelectedField 147
GetSelectedPageThumbnails 148
GetSelectedWidget 149
GetSelectionState 149
HighlightSelection 150
HighlightTextByFile 150
InsertEmptyPages 151
InsertPages 152
IsOperationGranted 153
Print 154
RemoveSecurity 155
RotatePages 155
Save 156
SaveAttachment 157
SelectAllText 157
SelectAnnot 157
SelectField 158

SummarizeAnnots ... 159
Pages ... 160
<Methods> ... 161
GetSelectedRanges ... 162
<Item> ... 162
<Methods> ... 163
GetAnnotName ... 163
GetAnnotRect ... 164
GetAnnotsCount ... 164
GetAnnotType ... 165
MovePointToScreenPoint ... 166
PagePointsToView Points ... 166
TranslateScreenPoint ... 167
View PointsToPagePoints ... 168
Text ... 168
<Methods> ... 169
AddLink ... 169
Get ... 170
GetQuads ... 170
GetSelected ... 171
GetSelectedRanges ... 172
Highlight ... 172
Select ... 173
StrikeOut ... 174
Underline ... 174
Chars ... 175
<Item> ... 176
Style ... 176
Font ... 176
Words ... 177
<Item> ... 177
Lines ... 178
<Item> ... 178
Form ... 179
HighlightFields ... 179
View ... 180
Export ... 181
Image ... 182
Find ... 183
Options ... 183
Forms ... 183
HighlightFields ... 184
General ... 184
International ... 185
Notifications ... 186
BeforeCloseDoc ... 187
BeforeSaveDoc ... 187
ContextMenu ... 188
DocClosed ... 188
DocSaved ... 189
FieldChanged ... 189
Keyboard ... 190
Mouse ... 191
Print ... 192
Selection ... 193
TextEditor ... 194
Operations ... 195
DeletePages ... 195
ExtractPages ... 196

InsertEmptyPages ... 197
InsertPages ... 197
New Document ... 198
FromBlank ... 198
FromImages ... 199
Paper ... 199
Layout ... 200
Graphics ... 200
Labels ... 201
FromText ... 202
Paper ... 202
Layout ... 203
ColSep ... 203
FileHeader ... 203
Line ... 204
RotatePages ... 204
SummarizeAnnots ... 205
Output ... 206
PDF ... 207
RTF ... 207
TXT ... 208
HTML ... 208
PageDisplay ... 208
Performance ... 210
Print ... 210
ScaleSimple ... 212
Prompts ... 212
ConfirmDocumentIncSave ... 213
ConfirmDocumentSave ... 214
ConfirmDropFile ... 215
ConfirmFileReplace ... 217
ConfirmLaunchFile ... 218
ConfirmOpenSite ... 219
CreateFolderError ... 220
EnterDocumentPassword ... 221
FileWriteError ... 222
Search ... 223
Options ... 223
What ... 224
Where ... 224
Tools ... 225
Area ... 226
Arrow ... 227
Callout ... 227
Cloud ... 227
Distance ... 228
Eraser ... 228
FileAttachment ... 228
Highlight ... 229
Line ... 229
Link ... 229
Loupe ... 230
Oval ... 230
Pencil ... 230
Perimeter ... 231
Polygon ... 231
PolyLine ... 232
Rect ... 232
Snapshot ... 232

Stamp ... 233
StickyNote ... 233
StrikeOut ... 233
TextBox ... 234
TypeWriter ... 234
Underline ... 235
View ... 235
Colors ... 237
Multiply Referred Objects ... 238
Bars ... 238
<Item> ... 238
Border ... 239
Panes ... 239
<Item> ... 239
RectangleF ... 240
Text Format ... 240
Char ... 241
Para ... 241
UI ... 242
Labels ... 243
<Item> ... 243
Quad ... 243
Quads ... 244
Values ... 244
Blend Modes ... 246
Booleans ... 247
Border Effects ... 247
Border Types ... 247
Code Pages ... 247
Color Management Engines ... 248
Colors ... 248
Command States ... 248
Comment Subject Modes ... 249
Content Monitor States ... 249
Context Menu User Choices ... 249
Dash Types ... 250
Document Bars ... 250
Document Initial View Modes ... 250
Document Interfaces ... 251
Document Panes ... 251
Document Save Methods ... 252
Document Save Modes ... 252
Document SaveAs Destination Types ... 252
Duplex Printing ... 253
Export Modes ... 253
Export to Image Modes ... 253
File Attachment Icon Types ... 254
Highlight Form Fields Masks ... 254
Image Align Types ... 254
Image Conversion Types ... 255
Image Scale Types ... 255
Image Types ... 255
Keyboard Notifications Filter Flags ... 256
Line Ending ... 256
Link Highlight Mode ... 257
Main Bars ... 257
Main Panes ... 258
Mouse Notifications Filter Flags ... 258
Name Generation Macros ... 259

New Paragraph Modes ... 259
Pages Layouts ... 260
Pages Magnification Modes ... 260
Pages Magnifications ... 260
Paper Modes ... 261
Paper Orientation ... 261
PDF-Specification Versions ... 261
Print Notifications Filter Flags ... 262
Print Scale Types ... 262
Print Specials ... 263
Print Text as Curves ... 263
Range Filters ... 263
Range Types ... 264
Registry Roots ... 264
Rotation Direction ... 265
Search Modes ... 265
Selection Notifications Filter Flags ... 265
Shortcut Key Types ... 265
Shortcut Modifiers ... 266
Sticky Note Icon Types ... 266
Summarize Annotations Group Types ... 266
Summarize Annotations Output Types ... 267
Summarize Annotations PDF Layouts ... 267
Text Align ... 267
Text Editor Notifications Filter Flags ... 268
Text File Placing Modes ... 268
Text Rendering Mode ... 268
Tools ... 269
User Choices ... 269
User Interface Languages ... 270
Simple Notifications ... 271
Global ... 271
Select Tool ... 271
Text Editor ... 272
3 JavaScript Support ... 272
Annotation ... 273
app ... 273
Bookmark ... 274
console ... 275
Doc ... 275
event ... 277
Field ... 278
FullScreen ... 278
global ... 279
identity ... 279
Link ... 279
OCG ... 280
Template ... 280
util ... 280
4 Enumerations ... 281
PXCVA_DocumentSaveFlags ... 281
PXCVA_EventTypes ... 282
PXCVA_Flags ... 283
PXCVA_ViewObjectTypes ... 283
5 Simple ActiveX Control ... 284
IPDFXCpreview ... 284
Methods ... 285
Print ... 285

Properties ... 285
Allow Accelerators ... 285
LockedView ... 286
ReadOnly ... 286
SettingsURL ... 287
Src ... 287

Part III How To Use 288

1 How to Open a Document? ... 288
2 How to Print a Document? ... 289
3 How to Save a Document? ... 290
4 How to Close a Document? ... 291
5 How to Disable a Command? ... 292
6 How to Enumerate Characters in Document? ... 293
7 How to Extract Text from Document? ... 295
8 How to Summarize Annotations from Document? ... 296

Part IV Tracker Software Products 297

1 Contact Us ... 298
2 Products ... 298

Index 300

1 Important - Please Read

The PDF-XChange Viewer SDK - Version 2.x.

Please note: This is V2.x of the Viewer SDK and your existing V1.x License codes need to be replaced.

You will get a free upgrade if :
You purchased any Version 3 or 4 PDF-XChange or Tools SDK on or After Jan 1st 2007, or if you purchased the PDF-XChange Viewer SDK Version 1.x as a stand alone product.

You will need to pay for his upgrade if :

If your PDF-XChange Viewer SDK license was issued as a result of you owning any Version 3 PDF-XChange or Tools SDK purchased before Jan 1st 2007.

You will need to sign a distribution statement.

Before any new Version 2 Licenses are issued you will need to sign a statement stating how many PDF-XChange Viewer SDK Client Distribution License Packs (CDLP's) have been issued as a result of use of the Viewer SDK in your end user applications - no license codes will be issued without this.

General Info

The **PDF-XChange Viewer SDK** is available as both a simple DLL library to rasterize PDF pages and display them within a developer's application, or as a fully featured ActiveX representation of our highly popular and advanced End User PDF-XChange Viewer which allows developers to embed extensive viewing and manipulation functionality directly in their application Windows.

This file/document deals with the ActiveX Version and its properties and methods - please see the help file : **PXCView36SSDK_Help.chm** included in this installation for details on the *PDF-XChange Simple DLL Viewer SDK*.

As with all of our products the evaluation version of the PDF-XChange Viewer is fully functional and also the version you should use in your developed applications - all that distinguishes live use from evaluation use, is that for live use you must provide your purchased license strings in your project code when compiling your application - this ensures our 'trial' watermarks do not appear in the top corners of all pages from being displayed or printed when used.

Please - do not buy immediately!

We are confident that all of our products offer unrivalled value and functionality. We offer fully functional, no timeout, versions **of all our Tools** for download and evaluation **- we therefore strongly recommend** that developer's not only use the evaluation to test out the functionality available before buying - but even once they have decided to purchase in principle - wait until they have fully completed integrating the desired functionality into their first production application, ready to deliver to their first client - before purchasing and **only then should you buy!**

By doing so, we mutually ensure your 100% satisfaction before parting with any money and that we have only contented and happy developer clients. F**or this reason we do not offer refunds,** as you are able to use the evaluation version without timeout or restriction (save the trial watermarks previously mentioned on all output) - and there should be no dissatisfaction with the product purchased.

We also recommend that developers install the general and FREE **End user release** of the **PDF-XChange Viewer** (including help files and localised language files) to familiarise themselves with the functionality available to them and their clients by using the PDF-XChange Viewer both within and if you

wish - external to your applications.

Please also note that the features marked as 'PRO' and charged for in the end user release - when used within a developers application and a license has been paid for - function without limitations or impediment - it is only if used externally of the developer's application that the PRO features woudl revert to 'evaluation' mode and not be freely useable without dispalying/printing the evaluation watermarks etc.

This is available for download (and may be freely distributed if you wish) directly from this link: click here.

Upgrade Information

We constantly strive to improve and extend the functionality available for our all of our products and tend not to wait for major version upgrades before releasing improvements - these are released incrementally as they become available.

Typically we release new versions with bug fixes and functionality improvements approx every 1-2 months and recommend that all our clients check our web site frequently for updates - if not monthly then at least quarterly, particularly if you discover a problem. Please check for a new release before contacting us for assistance - with many, many millions of active users - the chances are we are aware of problems before you locate them and whenever possible will issue a fix without delay.

Upgrade Policy

When you purchase any of our products, your are currently entitled to all major and minor upgrades for a minimum of 12 months after the purchase date - after which time you are entitled to all minor updates free of charge, up to - but excluding - the next Major version update.

During 2008/9 we will also be introducing a maintenance option, payable annually, which will entitle subscribers to all ongoing updates irrespective of whether they are Major or Minor in nature during the currency of their subscription - we will make all clients aware when this is introduced and any changes to our upgrade policy this may result in at that time, however, naturally, no purchasers will be deprived of the minimum 12 months of free upgrades - irrespective of how our T&C's may subsequently alter.

Support

All support is provided free - whether you are using an evaluation, or licensed release of our products.

As any developer will recognize - email is increasingly an issue with regards the huge volumes of spam being circulated and the resulting problems this causes with antivirus, anti-spam, filtering etc.

We are no different and receive 10,000's of emails each day unrelated to our business - **for this reason we request that all support issues** are made via our User Forums whenever possible - you will need to register (free) and acknowledge the email sent to your inbox before you can View messages or Post.

Our forums are monitored and all posts answered by our own technical support staff and also provide a useful historic repository of previous Q&A's and FAQ's.

There are also Forums specifically created for the posting of your 'wish list' should you have any comments or ideas you wish to share with us.

If you do need to email us for any reason - please see our Contacts page for the relevant contact info.

Important

Any attachment sent by email or posted to our forums MUST be wrapped in either a ZIP, 7z or RAR archive - failure to do so will result in your message being stripped of its attachment and we will not receive!

1.1 Redistribution Information

To redistribute your own application to your end users, the following components should be included in your installation and you must also register the components as outlined below:

PDFXCview.exe
PDFXCviewAx.dll
Resource.dat

Note:

There are both 32 and 64 bit versions of the PDF-XChange Viewer control and you should accommodate both in your distribution – please see the appropriate SDK installation folder to locate the 32/64 bit versions provided. For details about this, see ActiveX Control Installation.

Important: The PDF-XChange Viewer SDK requires Windows 2000 or later!

See Also

IPDFXCview::GetVersionInfo

1.2 ActiveX Control Installation

To install the PDF-XChange Viewer ActiveX manually or via your applications installer:

To install manually:

- Run **PDFXCview.exe** with the "**/RegServer**" parameter.
- Run **regsvr32.exe** with the "**PDFXCviewAx.dll**" parameter.

To uninstall manually:

- Run **PDFXCview.exe** with the "**/UnregServer**" parameter.
- Run **regsvr32.exe** with the "**PDFXCviewAx.dll**" and "**/u**" parameters.

To install manually under x64 systems:

for 32-bit clients:
PDFXCview.exe(32) **/RegServer**
regsvr32 PDFXCviewAx.dll(32) **/s**
or:
PDFXCview.exe(64) **/RegServer** (the **PDFXCview.tbl**(32) should be in the same folder as PDFXCview.exe(64))
regsvr32 PDFXCviewAx.dll(32) **/s**
for 64-bit clients:
PDFXCview.exe(64) **/RegServer**
regsvr32 PDFXCviewAx.dll(64) **/s**
for 64-bit and 32-bit clients both:
PDFXCview.exe(64) **/RegServer** (the **PDFXCview.tbl**(32) should be in the same folder as PDFXCview.exe(64))
regsvr32 PDFXCviewAx.dll(32) **/s**
regsvr32 PDFXCviewAx.dll(64) **/s**

To install automatically:

use our *.exe installer (in silent mode optionally):
PDF-XChange Viewer ActiveX SDK - Distribution

for MSI Installers you may use special *MSM* module to include our PDF-XChange Viewer ActiveX into your installation process:
MSM for 32-bit clients,
MSM for 64/32-bit clients.

Registration-Free Deployment

You may use the PDF-XChange Viewer ActiveX control without the standard COM-integration usually required on a target system (e.g. without registering an ActiveX).

To briefly explain, on Windows XP and later any client application can use any external COM-component even if it is not registered on the target system (i.e. without the requirement to enter basic info about the COM-component in the system registry).

To achieve this you are required by Windows to specify as a substitute, 2 specific files: **Server.X.manifest** and **Client.exe.manifest**, both files are XML formatted files.

The **Server.X.manifest** contains information about the external COM-component(s) to be used by the Client.exe application.

The **Client.exe.manifest** contains a brief description about your **Client.exe** assembly and any dependency between the Client and Server architecture. In the PDF-XChange Viewer ActiveX SDK example folder: **"Registration-Free COM"** you will find examples of this method of use: **PDFXCviewAx.X.manifest** file (for the Server side) and **Client.exe.manifest** - a *template* manifest for any client application (Client side).

So, to use this method, you should:
1. Copy both files from this SDK-folder to your program folder.
2. Rename the Client.exe.manifest to <YourProgramName>.exe.manifest.
3. Open the <YourProgramName>.exe.manifest, replace **name="Client"** to **name="<YourProgramName>"** and save changes.
4. Check your program folder, it should contain:
    <YourProgramName>.exe
    <YourProgramName>.exe.manifest
    PDFXCviewAx.X.manifest
    PDFXCviewAx.dll
    PDFXCview.exe
    resource.dat

Having followed the above instructions <YourProgramName>.exe application will be able to load and use the Viewer control without the usual standard COM-integration on a target system.

Tips:
When developing your application, before adding our control to your form, temporarily install using our standard method rather than the 'registration free' (it's described above, for example you can do so by running command line switches: "PDFXCview.exe /RegServer", "regsvr32 PDFXCviewAx.dll /s").

Once done you can simply add our ActiveX control to your form like any other COM-component.

Having done the above specify the manifests as described and uninstall our control from your system ("PDFXCview.exe /UnRegServer", "regsvr32 PDFXCviewAx.dll /u /s") - our control will now continue to function using the specified manifests.

Now, when installing on your end users systems - your installer can specify the required manifests only and the standard COM installation is no longer a requirement.

For details about this technology: http://msdn.microsoft.com/en-us/library/ms973913.aspx

See Also

IPDFXCview::SetDevInfo

1.3 Licensing

The PDF-XChange Viewer SDK

Understanding the License types available

Allows the developer to incorporate PDF viewing Capabilities within their software applications. Supports PDF format specifications up to 1.7 (Adobe 8).

The Simple DLL SDK provides the developer with set of functions to create a means to View/Print PDF files within a window embedded in their software application. Both the GUI design and means of employing the provided functionality is very much left in the developer's hands and no Tracker branding is visible within the Developer's product.

The ActiveX SDK option is a fully featured, ready to incorporate facsimile of the End User PDF-XChange Viewer, again embedded within a Window of the application, the developer is able to hide and disable most of the controls and functions within the SDK and some GUI tailoring is possible - but Tracker Software branding is present and required from the 'About' Window of the embedded viewer. The ActiveX is also more feature rich than the Simple DLL method.

Developer's owning existing PDF-XChange or Tools Developer SDK's are provided with the following free distribution rights to end users of their software applications:

PDF-XChange/Tools SDK Type	Viewer Client Distribution Licenses Included
PDF-XChange Viewer Base SDK	25,000
PDF-XChange PRO SDK	7,500
PDF-XChange Drivers API SDK	2,500
PDF-Tools SDK	2,500

Once the licenses provided with any of the above Developers SDK's have been used - prior to continuing to distribute an application utilising the PDF-XChange Viewer, the Developer must purchase an additional client distribution License Pack (CDLP) available in 50K, 100K, 250K or 1 Million Client License pack's. For information regarding Royalty free licensing please contact oemsales@tracker-software.com.

If a developer does not own any of the above SDK's, the purchase of a PDF-XChange Base Viewer SDK is required prior to purchasing an Client Distribution. License packs.

The PDF-XChange Viewer SDK is not a Royalty Free product as standard - however special rates may be negotiated for large Client Distribution License packs.

Important: either a **PDF-XChange Base Viewer SDK** or one of our other **PDF-XChange/Tools SDK's** must be owned - prior to additional CDLP License packs being purchased.

Please note prices subject to change - please check our web site for the latest available costing's.

1.3.1 License Text

License Agreement for the PDF-XChange Viewer (SDK) from Tracker Software Products (Canada) Ltd 2006-2009.

PRINTED BELOW IN ITS ENTIRETY IS THE LICENSE AGREEMENT GOVERNING YOUR USE OF THE SOFTWARE.
PLEASE READ THE LICENSE AGREEMENT.

IMPORTANT

TRACKER SOFTWARE PRODUCTS LTD. IS WILLING TO LICENSE THE ENCLOSED SOFTWARE TO YOU ONLY UPON THE CONDITION THAT YOU ACCEPT ALL OF THE TERMS CONTAINED IN THE LICENSE AGREEMENT PRINTED BELOW. PLEASE READ THE TERMS CAREFULLY BEFORE OPENING THE PACKAGE CONTAINING THE DISKETTE(S)/CD-R(S), Electronic File OR CLICKING THE ACCEPT BUTTON DURING INSTALLATION, AS SUCH CONDUCT INDICATES YOUR ACCEPTANCE TO ALL OF THE TERMS OF THIS LICENSE AGREEMENT. IF YOU DO NOT AGREE TO THE TERMS, TRACKER SOFTWARE PRODUCTS LTD IS UNWILLING TO LICENSE THE SOFTWARE TO YOU, IN WHICH CASE YOU MUST IMMEDIATELY RETURN THE PACKAGE AND ALL ACCOMPANYING MATERIAL TO TRACKER SOFTWARE PRODUCTS LTD. OR YOUR AUTHORIZED DEALER FOR A FULL REFUND.

This License Agreement ("Agreement") is a legal agreement between Tracker Software Products (Canada) Ltd, (Tracker), a Company registered in Canada, principally located at 466 Trans Canada Highway, Duncan, BC. V9L3R6. Canada, and you, the user ("Licensee"), and is effective the date Licensee opens the package containing the diskette(s)/CD-R(s) or otherwise uses the enclosed software product.

This Agreement covers all materials associated with Tracker's PDF-XChange Viewer SDK developer's toolkit products, both 'Simple DLL' and 'ActiveX' based options - including the enclosed software product ("Software").

1. GRANT OF DEVELOPMENT LICENSE

TRACKER grants Licensee a non-exclusive, non-transferable, worldwide license for one (1) programmer to install the Software on a single personal computer and use the Software and one copy of the associated user documentation contained in the accompanying user manual, "online" help and Acrobat files ("Documentation") in the development of End User software application's as contemplated in section 2 below (herein, the "Application Software"). If additional programming seats are needed, Licensee should contact TRACKER for discounted license pricing. The license granted hereunder applies only to the designated version of the enclosed Software. If the Software is an upgrade or cross grade, it, and the product that was upgraded/cross graded constitute a single copy of the Software for purposes hereof and the new version and product that was upgraded/cross graded cannot be used by two people at the same time.

2. END USER APPLICATION

The Application Software developed by Licensee must be an "End User Application." An "end user application" is a specific application program that is licensed to a person or firm for business or personal use and not with a view toward redistributing the application or any part of the application, and may be either an application that is used by Licensee internally, or an application that is commercially distributed to end users for their use. A user of an end user application may not modify or redistribute the application and may not copy it (other than for archival purposes). Licensee's license agreement covering the Application Software must contain restrictions prohibiting redistribution, modification and copying of the Application Software. The license rights hereunder do not apply to development and deployment of software products such as Printer Drivers, ActiveX controls, plug-ins, authoring tools, development toolkits, compilers, operating systems and also software products where a significant function is to generate 'PDF' format files (as defined by ISO Standards body) and other file formats from 3rd party software applications not developed by the licensee, indirectly or otherwise, - such as Microsoft's 'Office' suite and component applications other than for the purpose of creating and then storing such files within a structured application for the archival and management of documents, that is developed by the licensee and any other software not falling within the definition of an end user application Further, Licensee may not, under any circumstances, create a competing software

application to Trackers own “PDF-XChange Viewer” for End users, or for which a significant intended purpose is the viewing or manipulation of PDF format files, without first requesting Tracker to specifically agreeing to the creation and distribution of such a product. If Licensee wishes to develop a product outside the scope of this license, Licensee should contact TRACKER’S OEM Sales department to see if a special license is available.

3. GRANT OF DUPLICATION AND DISTRIBUTION LICENSE

The Software includes certain runtime libraries and files intended for duplication and distribution by Licensee within the Application Software to the user of Application Software ("Redistributables"). The Redistributable components of the Software are those files specifically designated as being distributable in the "Files to be Included with Your Application" section of the Online Help file, the terms of which are hereby incorporated herein by reference. Licensee should refer to the Documentation and specifically the "Online Help" file for additional information regarding the Redistributables. Under TRACKER’S copyright, and subject to all the restrictions and conditions set forth in this Agreement and the Documentation, TRACKER hereby grants Licensee (and only Licensee) a non-exclusive, non-transferable, worldwide license to reproduce exact copies of the Redistributables and include such files in the Application Software, and to deploy the Application Software internally and/or distribute the Application Software, directly or through customary distribution channels, to end users to the limits prescribed below in Section 4, "Duplication and Distribution of Royalty Bearing Versions " below.) If Licensee wishes to use an OEM who will modify the Application Software and copy it, Licensee must first obtain an OEM distribution license from TRACKER or must require the OEM to obtain a license from TRACKER. Duplication or Redistribution of the Application Software, or any portion thereof, by the users of the Application Software, without a separate written redistribution license from TRACKER is prohibited. If the enclosed Software is packaged "For Evaluation Only," no right to copy and/or distribute the Redistributables is granted. No rights to copy or redistribute the Application Software are granted until such time as Licensee has properly licensed and registered the Software with TRACKER and otherwise complied with this Agreement. Unless otherwise agreed in writing by Tracker,

4. DUPLICATION AND DISTRIBUTION OF ROYALTY BEARING VERSIONS OF THE SOFTWARE

The enclosed software is a Royalty Bearing software development kit and may not be distributed Free of Royalties - your initial purchase of one of the software products detailed below includes the right (subject to your acceptance of the terms and conditions of this agreement) to embed within your software application the PDF-XChange Viewer SDK by accessing either the DLL or ActiveX based versions of the PDF-XChange Viewer SDK, subject to an appropriate purchase and distribute a specified number of user licenses to your end user software application clients. When this limit is reached you must purchase additional distribution licenses prior to any further distribution or remove the software from your application prior to further distribution of your application. When calculating your distribution of licenses, each user having access to use of your application incorporating this software must be accounted for individually, Server, Concurrent and Site licensing models are not acceptable or applicable for this purpose.

The following Tracker developer kits include limited distribution rights to the software for an initial specified number of Client Desktop License Packs (CDLP’s) as detailed below, this is an indication only and the actual specified and agreed number of licenses may be different and is detailed and accepted by the parties when signed on the final page of this document.

PDF-XChange Viewer Basic SDK Pack – Initial max distribution included: 25,000 Single user Licenses(available in a variety of predetermined or negotiated license packs)
PDF-XChange PRO SDK Version 4 – max distribution included: 7500 Single user Licenses
PDF-XChange Drivers API SDK Version 4 – max distribution included: 2500 Single user Licenses
PDF-XChange Tools SDK Version 4 – max distribution included: 2500 Single user Licenses

PDF-XChange Viewer SDK extended License packs (CDLP’s) are available for a variety of volume requirements and on a Royalty Free basis – please see our web site or contact sales@tracker-software.com for more detailed information.

No duplication or distribution rights are granted hereunder with respect to the Royalty Bearing Versions to enable live use and distribution of your application(s) using this software until Tracker have received from you, a copy of this agreement **with each page initialled** and the last page signed acknowledging your understanding and acceptance of all the terms of this agreement, **only then will you receive your license unlock codes** enabling use other than for evaluation purposes.

Licensee agrees to account on request by TRACKER for all applications sold or distributed by Licensee or its subsidiaries incorporating the software since its first inclusion in the products of the Licensee - within 28 days of such request having been received from Tracker to the Licensee's contact information as provided (either by post or email). In the event the licensee does not respond, has exceeded the limits detailed within this agreement or any dispute regarding license volumes & payment, Tracker Software Products Ltd may appoint a qualified Auditor to authenticate the records of the Licensee to establish the validity and the Licensee agrees to make all records available pertaining to the Licensee's accounting and other related information, without exception, on written request within 24 hours of receipt of such a request during normal working hours. In the event that such an audit reveals any material inaccuracy in the reporting of the licensee sales and royalty liabilities to TRACKER - Licensee shall:

- Make full payment to TRACKER of all outstanding royalty liabilities within 7 days of demand
- Pay in full all fees and associated costs of the audit howsoever arising
- Immediately cease all sales of all products containing Tracker Software Products Ltd's Toolkits or intellectual property until guarantees of the future reliability of the Licensee's reporting to the satisfaction of Tracker Software Products Ltd are provided.
- **These remedies shall not restrict or limit** such other avenues for compensation or damages as may be allowed by the laws of Canada – but set forth the minimum remedy that Tracker Software Products shall be entitled to, without delay, further negotiation or legal recourse.

5. OTHER RESTRICTIONS

The licenses granted under this Agreement are expressly conditioned upon Licensee's compliance with all the terms and conditions of this Agreement. Licensee may not use, copy, rent, lease, sell, sublicense, assign or otherwise transfer the Software except as expressly provided for in this Agreement. Licensee may make a reasonable number of archival copies of the Software. Except for the Redistributables, Licensee shall not distribute any files contained in the Software, including without limitation, .EXE, DLL, CLW, .INC, .TPL, .CHM, .DRV, .LIB, .H, .MAK, .DEF, .TXT, .PDF or .HLP files. Licensee shall not reproduce, copy or transfer any Documentation, except Licensee may use the sample source code examples contained in the Documentation for the purpose of developing the Application Software. Upon TRACKER'S request, Licensee agrees to send TRACKER one demonstration copy of the Application Software for evaluation and assessment. Any distributor or reseller of Application Software appointed by Licensee must be subject to a binding agreement that includes provisions no less protective of TRACKER'S intellectual property rights in the Software as it is protective of Licensee's rights in its own software. Licensee acknowledges that the Software, in source code form, remains a confidential trade secret of TRACKER and/or its suppliers and therefore Licensee agrees that it shall not modify, decompile, disassemble or reverse engineer the Software or attempt to do so except as permitted by applicable legislation. Licensee agrees to refrain from disclosing the Software (and to take reasonable measures with its employees to ensure they do not disclose the Software) to any person, firm or entity except as expressly permitted herein. Specifically, Licensee will not disclose or publish any license or unlock codes or instruction sets provided by TRACKER relating to the Software. If Licensee wishes to use the Software in a manner specifically or generally prohibited by this Agreement, Licensee should contact TRACKER'S OEM department to determine whether a special license may be required/obtained.

6. Use in Evaluation versions of Licensee Developer's End User Applications.

Where the Licensee provides a limited use evaluation version of their End User software – distribution of the PDF-XChange Viewer SDK redistributable components shall not count towards the licensee's allowed limited redistribution totals - provided that:

1: The functionality provided by the PDF-XChange Viewer SDK will cease on expiry of the evaluation period.

2: The evaluation period is no longer than 90 days – otherwise it must count towards the overall limited redistribution rights provided by the clients purchased license and may not be provided under the umbrella of this evaluation concession.
3: In the event that the Licensee allows some functionality of their application to continue after the evaluation period expires – this will not include the PDF-XChange Viewer SDK functions - otherwise it must count towards the overall limited redistribution rights provided by the clients purchased license and may not be provided under the umbrella of this evaluation concession.

If the Licensee is unable to ensure the above limited use restrictions are adhered to – Licensee may not incorporate the PDF-XChange Viewer SDK functionality in evaluation versions of their software applications – without accounting for each installation towards the total license count, allowed under the terms of this license.

7. PROPRIETARY RIGHTS; COPYRIGHT NOTICES

Except for the limited license granted herein, TRACKER, and its suppliers, retains exclusive ownership of all intellectual and proprietary rights (including all ownership rights, title, and interest) in and to the Software. Licensee agrees not to represent that TRACKER is affiliated with or approves of Licensee's Application Software in any way. Except as required hereby, Licensee shall not use TRACKER'S name, trademarks, or any TRACKER designation in association with Licensee's Application Software. The Application Software should contain the following copyright notice in the "About box" or if not the About box as a minimum, Developers License as provided with the End User Application: "Portions of this product were created using PDF-XChange & Image-XChange SDK's From Tracker Software Products Ltd ©2001-9, ALL RIGHTS RESERVED."

8. EXPORT LAW

Licensee acknowledges and agrees that the Software and Application Software may be subject to restrictions and controls imposed by the United States Export Administration Act, as amended (the "ACT"), and the regulations there under. Licensee agrees and certifies that neither the Software nor any direct product thereof (e.g. the Application Software) is being or will be acquired, shipped, transferred or re-exported, directly or indirectly, into any country prohibited by the ACT and the regulations there under or will be used for any purpose prohibited by the same. Licensee acknowledges that the Software may include "technical data" subject to export and re-export restrictions imposed by U.S. law. Licensee bears all responsibility for export law compliance and will indemnify TRACKER against all claims based on Licensee's exporting of the Application Software.

9. U.S. GOVERNMENT RESTRICTED RIGHTS LEGEND

Use, duplication or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 and FAR 52.227-19, as applicable. Manufacturer/Contractor is Tracker Software Products (Canada) Ltd, 466 Trans Canada Highway, Duncan, BC. V9L3R6. Canada.

10. TERM

The license granted hereby is effective until terminated. Licensee may terminate the license by returning the Software and Documentation to TRACKER, without refund, and destroying all copies thereof in any form. TRACKER may terminate the licenses if Licensee fails to comply with any term or condition of this Agreement or any corresponding duplication and distribution agreement for Printer Driver Products. Upon such termination, Licensee shall cease using the Software and cease using or distributing the Application Software containing the Redistributables. All restrictions prohibiting Licensee's use of the Software and intellectual property provisions relating to Software running to the benefit of TRACKER will survive termination of the license pursuant hereto. Termination will not affect properly granted end user licenses of the Application Software distributed by Licensee prior to termination, subject to the conditions further detailed in Clause15 below, titled : **LICENSE EXPIRY**.

11. EXCLUSION OF WARRANTIES

TRACKER and its suppliers offer and Licensee accepts the Software "AS IS." TRACKER and its suppliers do not warrant the Software will meet Licensee's requirements or will operate uninterrupted or error-free. ALL WARRANTIES, EXPRESS OR IMPLIED, ARE EXCLUDED FROM THIS AGREEMENT AND SHALL NOT APPLY TO ANY SOFTWARE LICENSED UNDER THIS AGREEMENT, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.

12. LICENSEE'S REMEDIES: LIMITATIONS

LICENSEE'S SOLE AND EXCLUSIVE REMEDIES AGAINST TRACKER ON ANY AND ALL LEGAL OR EQUITABLE THEORIES OF RECOVERY SHALL BE, AT TRACKER'S SOLE DISCRETION, (A) REPAIR OR REPLACEMENT OF DEFECTIVE SOFTWARE; OR (B) REFUND OF THE LICENSE FEE PAID BY LICENSEE.

13. NO LIABILITY FOR CONSEQUENTIAL DAMAGES

In no event shall TRACKER, or its suppliers, be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information or other pecuniary loss) arising out of use of or inability to use the Software, even if TRACKER or its dealer have been advised of the possibility of such damages. Some states do not allow the exclusion or limitation of certain implied warranties or the exclusion or limitation of incidental or consequential damages, in which case and to the extent such exclusion or limitation is not allowed, some of the foregoing limitations and exclusions may not apply to Licensee.

14. UPDATES AND UPGRADES

From time to time Tracker at its sole discretion, will release updates and upgrades incorporating bug fixes and new features, during the first 12 months after purchase you will receive free of charge any **Minor Releases** issued relevant to the version purchased (e.g. if you purchased Version 2.x, then all version 2.x releases issued will be provided free and so on) provided all other terms and conditions of this agreement have been compiled with.

Major Releases (e.g. from Version 2.x to Version 3.x etc) will be provided free of charge for no less than 3 months after the initial purchase after which time Tracker reserves the right to apply a fee for access to Major upgrades.

Further when issuing updates of any nature, Tracker reserves the right to disable previous licensing codes from being useable in the new release, to 'trigger' a distribution statement from Developer's, once the Distribution Statement and any revised license has been returned, and always providing the Developers distribution entitlements have not been exceeded – new license codes will be released. An example distribution statement is provided as an attachment to this document. Failure to provide a distribution statement when required is a material breach of this Agreement and renders the Developer liable to the remedies detailed under section 4 of this agreement.

15. LICENSE EXPIRY:

This Expiry condition is only relevant where the licensee is using the PDF-XChange Viewer SDK functionality as a result of the purchase of **PDF-XChange PRO SDK, PDF-XChange Drivers API SDK or PDF-Tools SDK** and the distribution limit is specified as being 7500 CDLP's or less. **IT DOES NOT IN ANY WAY AFFECT** any licensee using the PDF-XChange Viewer SDK if that Licensee has specifically purchased the PDF-XChange Viewer SDK and 25,000 or more CDLP's for distribution to end user client desktops.

On the event of a Major version Upgrade release to the PDF-XChange Viewer SDK , PDF-XChange PRO SDK, PDF-XChange Drivers API SDK or PDF-Tools SDK, all rights to continue to distribute applications using the PDF-XChange Viewer SDK functionality within developed application shall cease, such rights having been deemed expired and terminated as a consequence of the new Major release

event.

Should the developer wish to continue to distribute and utilise the functionality within developed applications, the purchase of an appropriate upgrade allowing the continued use and distribution of Viewer SDK enabled applications is required, should the developer not wish to purchase such an upgrade – the functionality provided must be removed from the developers published applications within 28 days of such a release and further distribution of any products containing the provided functionality halted immediately. Existing end user installations of the Developers applications **shall not be affected** , until such time as the developer chooses to release updated versions of their own application to such end user installations, upon which any updated application the developer may provide to clients must also be provided excluding any PDF-XChange Viewer SDK functionality, unless a paid for, appropriate upgrade has been purchased which would allow this.

16. GENERAL

This Agreement shall be interpreted, construed, and enforced according to the laws of Canada. In the event of any action under this Agreement, the parties agree that courts located in Canada will have exclusive jurisdiction and that a suit may only be brought in Canada, and Licensee submits itself for the jurisdiction and venue of the courts located in Canada. This Agreement constitutes the entire agreement and understanding of the parties and may be modified only in writing signed by both parties. No officer, salesman, or agent has any authority to obligate TRACKER by any terms, stipulations or conditions not expressed in the Agreement. All previous representations and agreements, if any, either verbal or written, referring to the subject matter of this Agreement are void. If any portion of this Agreement is determined to be legally invalid or unenforceable, such portion will be severed from this Agreement and the remainder of the Agreement will continue to be fully enforceable and valid. This Agreement, and the rights hereunder, may not be assigned by Licensee, whether by oral or written assignment, sale of assets, merger, consolidation or otherwise, without the express written consent of TRACKER. Licensee agrees to be responsible for any and all losses or damages arising out of or incurred in connection with the Application Software. Licensee agrees to defend, indemnify and hold TRACKER harmless from any such loss or damage, including attorney's fees, arising from the use, operation or performance of the Application Software or Licensee's breach of any terms of this Agreement. Licensee shall be responsible for paying all state and federal use, sales or value added taxes, duties or governmental charges, whether presently in force or which come into force in the future, related to the distribution and sale of the Application Software and will indemnify TRACKER against any claim made against TRACKER relating to any such taxes or assessments.

PDF-XChange Templates & Classes for Clarion for Windows (PDF-XChange-API/SDK customers only)
PDF-XChange API/SDK (PDF-XChange-API/SDK customers only)
PDF-XChange SDK Printer Driver (PDF-XChange-Print Driver customers only)
PDF-Tools SDK Templates & Classes for Clarion for Windows (PDF-Tools-API/SDK customers only)
Delphi Components for PDF-XChange and/or PDF-Tools SDK products.
All Demo/Evaluation components and examples for PDF-XChange and/or PDF-Tools SDK products.
Image-XChange SDK, PDF-XChange Viewer SDK

This agreement allows the Licensee to utilise and distribute the PDF-XChange Viewer SDK subject to the terms and conditions detailed above to the max number Client Distribution License Packs (CDLP's) specified (see below for CDLP's provided), each client desktop installed to, shall count as 1 CDLP having been used, after which, additional CDLP License packs will be required to continue use and distribution of any application created/distributed that contains the PDF-XChange Viewer SDK functionality.

Volume of CDLP's purchased and distributable:

CDLP Volume entitlement: | | | | | | | | | | |

Please provided your Purchase receipt or Invoice number # where you paid for the product, to allow us to expedite validation of

your license entitlement: _____

Date on which this payment was made: _____

Accepted for and on Behalf of Licensor

Tracker Software products Ltd

Name_____

Tel/Fax: 001-250-597-1621/001-250-897-1623

Email: sales@tracker-software.com

Position_____

466, Trans Canada Highway, Duncan, BC. V9L 3R6. Canada

Authorised Signatory:

_____

Date:

_____

Accepted for and on Behalf of Licensee

Company Name_____

Name_____

Tel/Fax:_____

*Email:_____

Position_____

Address_____

_____

Authorised Signatory:

_____

Date:

_____

*Please ensure you provide us with contact info that is robust and durable – particularly your email address, as this will be our primary means of advising you of updates and changes to your license and your license codes – using email addresses provided by FREE providers such as Hotmail and Yahoo is not recommended and we may even reject such use in the interests of serving you better.

The License becomes effective and valid as soon as the agreed license fee is paid and this document is duly completed and signed by both parties. Please return once duly signed and completed to Tracker Software Products limited by Facsimile, email as an attachment or by mail for completion by Tracker. Fax : 001-250-597-1623 or sales@tracker-software.com.

1.4 Requesting your V2.x Upgrade license

The PDF-XChange Viewer SDK - Version 2.0.0042.7.

Please note : This is V2.0042 of the Viewer SDK and your existing PDF-XChange V1.x License codes need to be replaced - they will not function in this release.

You will get a free upgrade if:
You purchased any Version 3 or 4 PDF-XChange or Tools SDK on or After Jan 1st 2007, or if you purchased the PDF-XChange Viewer SDK Version 1.x as a stand alone product.

You will need to pay for his upgrade if:

Your PDF-XChange Viewer SDK license was issued as a result of you owning any Version 3 PDF-

XChange or Tools SDK purchased before Jan 1st 2007.

You will need to sign a distribution statement.

Before any new Version 2 Licenses are issued you will need to sign a statement stating how many PDF-XChange Viewer SDK Client Distribution License Packs (CDLP's) have been issued as a result of use of the Viewer SDK in your end user applications.

There are 3 distinct products available for the inclusion of PDF Viewing within the developer's Software application - either of the below maybe purchased individually - or a PRO SDK containing both items below as a 'bundled' option.

2 Reference

This reference contains the following sections:

- Interfaces
- Named Items
- JavaScript Support
- Enumerations
- Simple ActiveX Control

2.1 Interfaces

The ActiveX control supports the following interfaces:

IPDFXCview	This interface allows applications to implement an instance of the **PDF-XChange Viewer control**.
_IPDFXCviewEvents	This interface allows applications to receive extended event notifications from the **PDF-XChange Viewer control**.
IPDFXCsmartp	This interface represents a special object (Smart Point) for simplification and access acceleration to any objects or properties.
IPDFXCview2	This interface is **default**, based on the IPDFXCview and contains some additional features for simple usage of the control.
IPDFXCargs	This interface represents an array of data-objects. Use it for simplification creation/reading/changing of data-object which contains array of data(**SAFEARRAY**).

Control Identificators:

```
CLSID: {FE36F0F3-F082-41B7-9EED-772505A7C054}
ProgID: PDFXCviewAx.CoPDFXCview
```

2.1.1 IPDFXCview

This interface is the base interface of our ActiveX control (Microsoft ActiveX control). This interface is derived directly from **IDispatch**.

All methods/properties of this interface could be separated into the following groups:

- General Methods
- Document Methods
- Auxiliary Methods
- Properties

Interface Identificator:

```
GUID(IID_IPDFXCview): {6BBDAD78-4AA9-40B1-977A-1D2A459B49C3}
```

Requirements

OS Versions: Windows 2000 and later.
TypeLib: PDFXCviewAx.tlb

2.1.1.1 General Methods

Methods

Method	Description
ApplyAllCachedChanges	Applies all cached settings that were passed with the **PXCVA_NoApply** flag.
DiscardAllCachedChanges	Discards all cached settings that were passed with the **PXCVA_NoApply** flag.
DoVerb	This is the **main method** in the library. All other methods act as shortcuts to some specific functionality of this method.
GetProperty	Retrieves specified property.
GetVersionInfo	This method returns the version of client and server portions of ActiveX control.
LoadSettings	Loads all control settings saved to external storage.
RunJavaScript	Performs Java script.
SaveSettings	Saves all control settings to an external storage.
SetDevInfo	This method allows you to enter license information. Without this information, wherever the document is visible, a demo label will appear. If you print, export, save the document, the result will also contain the demo label.
SetProperty	Sets specified property.

See Also

Document Methods, Auxiliary Methods

2.1.1.1.1 ApplyAllCachedChanges

Apply all cached settings that were passed with the PXCVA_Flags::PXCVA_NoApply flag.

Syntax

```
HRESULT ApplyAllCachedChanges(VOID);
```

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

PXCVA_Flags::PXCVA_NoApply,
PDFXCview::DiscardAllCachedChanges,
Operations::ApplyAllCachedChanges

2.1.1.1.2 DiscardAllCachedChanges

Discard all cached settings that were passed with the PXCVA_Flags::PXCVA_NoApply flag.

Syntax

```
HRESULT DiscardAllCachedChanges(VOID);
```

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

PXCVA_Flags::PXCVA_NoApply,
IPDFXCview::ApplyAllCachedChanges,
Operations::DiscardAllCachedChanges

2.1.1.1.3 DoVerb

DoVerb is the **main method** in the library. Other existing methods act as shortcuts to some specific functionality of this method.

Syntax

```
HRESULT DoVerb(
       BSTR ObjectName,
       BSTR OperationName,
       VARIANT DataIn,
       VARIANT* DataOut,
       LONG Flags
);
```

Parameters

ObjectName
: [in] **BSTR** that specifies the object name as string identifier. For more information, see Named Objects. This argument can be NULL.

OperationName
: [in] **BSTR** that specifies the necessary command/action for object specified by *ObjectName*. For more information, see Named Operations. This argument can be NULL if *ObjectName* is not NULL.

DataIn
: [in] **VARIANT** that specifies the necessary input data depending on the object specified by *ObjectName* and operation specified by *OperationName*. This argument may be empty.

DataOut
: [out] Pointer to a **VARIANT** structure that receives output data depending on the object specified by *ObjectName* and operation specified by *OperationName*. This argument can be NULL.

Flags
: [in] **LONG** that specifies optional flags. For more information, see PXCVA_Flags. This argument can be 0.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of received error code you can use GetTextFromResult.

Remarks

If you want to pass more than one input arguments then you should pass them by *DataIn* through **SAFEARRAY** structure. If you want to pass only the first argument to *DataIn* (no other arguments or optional) then you can pass this argument directly, without placing it into **SAFEARRAY** structure.
For example (in pseudocode):

```
// show open dialog:
DoVerb(NULL, "OpenDocument", DataIn, DataOut, 0);
// or open document directly:
DoVerb(NULL, "OpenDocument",
DataIn("C:\Test.pdf"), DataOut, 0);
// or, if document is password protected:
DoVerb(NULL, "OpenDocument",
DataIn(SafeArray("C:\Test.pdf", "password")), DataOut, 0);
```

See Also

IPDFXCview::GetProperty, IPDFXCview::SetProperty, IPDFXCview::DoDocumentVerb, Named Objects, Named Operations

2.1.1.1.4 GetProperty

This simplified method retrieves property values. It acts as a shortcut to the DoVerb main method.

Syntax

```
HRESULT GetProperty(
       BSTR Name,
       VARIANT* DataOut,
       LONG Flags
);
```

Parameters

Name
[in] **BSTR** that specifies full name of the property as string identifier.
For more information, see Named Objects.

DataOut
[out] Pointer to a **VARIANT** structure that receives value of the property specified by *Name*.

Flags
[in] **LONG** that specifies optional flags. This argument can be 0.
Also you can pass the PXCVA_Flags::PXCVA_GetNamed to obtain the *named value* (as a simple string) of the property specified by *Name*.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

PXCVA_Flags::PXCVA_GetNamed,

IPDFXCview::SetProperty

2.1.1.1.5 GetVersionInfo

This method returns the version of client and server portions of ActiveX control.

Syntax

```
HRESULT GetVersionInfo(
       BSTR* ClientVersion,
       BSTR* ServerVersion,
);
```

Parameters

ClientVersion
[out] Pointer to a **BSTR** that lets you receive the string which represents a client's portion of version info (version of PDFXCviewAx.dll module).

ServerVersion
[out] Pointer to a **BSTR** that lets you receive the string which represents a server's portion of version info (version of PDFXCview.exe module).

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

If *ClientVersion* is not equal to *ServerVersion* then you have a dangerous and unstable mismatch of module versions on your system. In this case please re-register of AcitveX control, see ActiveX Control Registration.

See Also

ActiveX Control Registration,
IPDFXCview::SetDevInfo

2.1.1.1.6 LoadSettings

Load all control settings from an external storage method of your choosing - such as an .INI file or registry location etc.

Syntax

```
HRESULT LoadSettings(
       VARIANT DataIn
);
```

Parameters

DataIn
[in] **VARIANT** that specifies a valid storage source.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCview::SaveSettings,
Operations::LoadSettings

2.1.1.1.7 RunJavaScript

RunJavaScript excutes a Java Script. It acts as a shortcut to the DoVerb main method.

Syntax

```
HRESULT RunJavaScript(
       BSTR Script,
       BSTR* Result,
       LONG ID,
       LONG Flags
);
```

Parameters

Script
[in] **BSTR** that specifies the Java Script text. For more information, see Java Script Support.

Result
[out] Pointer to a **BSTR** that receives result text of the executed Java Script specified by *Script*. This argument can be NULL.

ID
[in] **LONG** that specifies the unique identifier of the opened document.
This argument can be 0 for the active document.
The specified document will be used as the target for the script, referred by ***this*** in the script. For more information, see Java Script Support.

Flags
[in] **LONG** that specifies optional flags. For more information, see PXCVA_Flags.
This argument can be 0.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

Java Script Support,
Operations::RunJavaScript

2.1.1.1.8 SaveSettings

Save all control settings to an external storage method of your choosing - such as an .INI file or registry location etc..

Syntax

```
HRESULT SaveSettings(
       VARIANT* DataInOut
);
```

Parameters

DataInOut
[in,out] Pointer to a **VARIANT** that specifies a valid storage destination.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCview::LoadSettings,
Operations::SaveSettings

2.1.1.1.9 SetDevInfo

SetDevInfo allows you to enter PDF-XChange ActiveX license information **for the evaluation limitations to be removed**.

Without this information, wherever the document is visible, a demo label will appear. If you print, export, save the document, the result will also contain the Tracker Software/PDF-XChange demo label until your application is recompiled with the License Key (your serial number) and the Developer's Code (Your Devcode) embedded appropriately in your project code.

Please note these License strings are unique to the PDF-XChange Viewer SDK and are not common to any other License strings provided for our PDF-XChange Drivers API, PDF-Tools SDK or PDF-XChange PRO SDK. This is because in most instances the Viewer SDK is not a Royalty FREE tool and requires you to read, complete and return your license to Tracker Software Products - prior to Your License strings being provided.

For more information on licensing see this page see this page and/or Contact us.

Syntax

```
HRESULT SetDevInfo(
       BSTR Key,
       BSTR Code,
);
```

Parameters

Key
[in] **BSTR** that specifies the private developer's string key.
The template of key is "XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX".
Code
[in] **BSTR** that specifies the private developer's string code.
For example: "For Developer"

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code you can use GetTextFromResult.

See Also

IPDFXCview::GetVersionInfo

2.1.1.1.10 SetProperty

This simplified method allows you to set property values. It acts as a shortcut to the DoVerb main method.

Syntax

```
HRESULT SetProperty(
       BSTR Name,
       VARIANT DataIn,
       LONG Flags
);
```

Parameters

Name
[in] **BSTR** that specifies full name of the property as string identifier.
For more information, see Named Objects.

DataIn
[in] **VARIANT** that specifies the new value of the property specified by *Name*.

Flags
[in] **LONG** that specifies optional flags. This argument can be 0.
Also you can pass the PXCVA_Flags::PXCVA_NoApply to cache the passed value only.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

PXCVA_Flags::PXCVA_NoApply,
IPDFXCview::GetProperty

2.1.1.2 Document Methods

Document Methods that apply to individual documents.

Method	Description
ActivateDocument	Activate the document.
CloseAllDocuments	Close all opened documents.
CloseDocument	Close the opened document.
DoDocumentVerb	This method is the simplified version of the **DoVerb** method for working with open documents.
ExportDocument	Export the opened document, with the parameters defined earlier by Objects::Export names section.
FlushDocument	Flush all of the user's changes for opened document.
GetActiveDocument	Returns the unique identifier of the active document.
GetDocumentID	Returns the unique identifier of the opened document which is specified by order index.
GetDocumentIndex	Returns the order index of the opened document which is

	specified by unique identifier.
GetDocumentProperty	This method returns property values of the specified document.
GetDocumentsCount	Returns number of all opened documents.
OpenDocument	Open a document by the specified URL, local file name, or by the open file dialog.
PrintDocument	Print the opened document, with the parameters defined earlier by Objects::Print names section.
SaveDocument	Save the document.
SetDocumentProperty	This method sets property values of the specified document.

See Also

Document Methods, Auxiliary Methods

2.1.1.2.1 ActivateDocument

Activates the opened document. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT ActivateDocument(
       LONG ID,
       LONG Flags,
);
```

Parameters

ID
[in] **LONG** that specifies unique identifier of the opened document.

Flags
[in] **LONG** that specifies optional flags. For more information, see PXCVA_Flags. This argument can be 0.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

This call has no effect in **SDI** (Single Document Interface) mode.

See Also

IPDFXCview::GetActiveDocument, IPDFXCview::GetDocumentsCount, Operations::ActivateDocument

2.1.1.2.2 CloseAllDocuments

Closes all opened documents. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT CloseAllDocuments(
       LONG Flags
```

```
);
```

Parameters

Flags
[in] **LONG** that specifies optional flags. For more information, see PXCVA_Flags.
This argument can be 0.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

If you want to deny all UI prompts and alerts when calling this method then you must specify the PXCVA_Flags::PXCVA_NoUI flag in *Flags*.

See Also

IPDFXCview::CloseDocument,
Operations::CloseAllDocuments

2.1.1.2.3 CloseDocument

Closes the opened document. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT CloseDocument(
       LONG ID,
       LONG Flags
);
```

Parameters

ID
[in] **LONG** that specifies unique identifier of the opened document. This argument can be 0 for the active document.
Flags
[in] **LONG** that specifies optional flags. For more information, see PXCVA_Flags.
This argument can be 0.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

Call for close the opened document, specified by `ID`. If you want to deny all UI prompts, alerts (confirm save changes dialog, write error dialog, etc.) when calling this method you must specify the PXCVA_Flags::PXCVA_NoUI flag into *Flags*.

See Also

IPDFXCview::OpenDocument,
Operations::CloseDocument

2.1.1.2.4 DoDocumentVerb

This method is the simplified version of the DoVerb method for working with open documents.

Syntax

```
HRESULT DoDocumentVerb(
       LONG ID,
       BSTR ObjectName,
       BSTR OperationName,
       VARIANT DataIn,
       VARIANT* DataOut,
       LONG Flags
);
```

Parameters

ID
[in] **LONG** that specifies the unique identifier of the opened document. This argument can be 0 for the active document.

ObjectName
[in] **BSTR** that specifies the name of the specific object contained within or associated to the specified document as string identifier. For more information, see Objects::Documents.

OperationName
[in] **BSTR** that specifies the necessary command/action for the document object specified by *ObjectName*. For more information, see Named Operations. This argument can be NULL if *ObjectName* is not NULL.

DataIn
[in] **VARIANT** that specifies the necessary input data dependent on the object specified by *ObjectName* and operation specified by *OperationName*. This argument may be empty.

DataOut
[out] Pointer to a **VARIANT** structure that receives output data dependent with the object specified by *ObjectName* and operation specified by *OperationName*. This argument can be NULL.

Flags
[in] **LONG** that specifies optional flags. For more information, see PXCVA_Flags. This argument can be 0.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

If you want to pass only one argument to *DataIn* (with no additional arguments, or all other arguments are optional) then you can pass this argument directly. However, if you want to pass more than one input argument, then you should pass them by *DataIn* using a **SAFEARRAY** structure.

Unlike the DoVerb, the name of object does not contain the document identification prefix (*"Documents[#4095]."* for example) because the document is specified by ID argument already. I.e., the call (in pseudocode):

```
DoVerb("Documents[#4095].Title", ...);
```

is equivalent to call:

```
DoDocumentVerb(4095, "Title", ...);
```

See Also

IPDFXCview::OpenDocument, IPDFXCview::PrintDocument, IPDFXCview::ExportDocument, IPDFXCview::FlushDocument, IPDFXCview::SaveDocument, IPDFXCview::CloseDocument, IPDFXCview::GetDocumentProperty, IPDFXCview::SetDocumentProperty, Objects::Documents

2.1.1.2.5 ExportDocument

Export the opened document, with the parameters defined earlier by **Export** names section. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT ExportDocument(
       LONG ID,
       LONG Flags
);
```

Parameters

ID
[in] **LONG** that specifies unique identifier of the opened document. This argument can be 0 for the active document.

Flags
[in] **LONG** that specifies optional flags. For more information, see PXCVA_Flags. This argument can be 0.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

Call for export the opened document, specified by *ID*. If you wanted to skip the export dialog then you must specify the PXCVA_Flags::PXCVA_NoUI flag into *Flags*.

See Also

IPDFXCview::PrintDocument,
Objects::Export,
Operations::ExportDocument

2.1.1.2.6 FlushDocument

Flush all of the user's changes for specified opened document. Call this method to finish editing of a document's objects and apply new changes. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT FlushDocument(
       LONG ID,
       LONG Flags
);
```

Parameters

ID

[in] **LONG** that specifies unique identifier of the opened document. This argument can be 0 for the active document.

Flags

[in] **LONG** that specifies optional flags. For more information, see PXCVA_Flags. This argument can be 0.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCview::SaveDocument,
Operations::FlushDocument

2.1.1.2.7 GetActiveDocument

Returns the unique identifier of the active document. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT GetActiveDocument(
       LONG* ID
);
```

Parameters

ID

[out] Pointer to a **LONG** that receives unique identifier of the active document.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCview::ActivateDocument, IPDFXCview::GetDocumentsCount,
Operations::GetActiveDocument

2.1.1.2.8 GetDocumentID

Returns the unique identifier of the opened document which is specified by order index. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT GetDocumentID(
       LONG Index,
       LONG* ID
);
```

Parameters

Index

[in] **LONG** that specifies the order index of the opened document.

ID

[out] Pointer to a **LONG** that receives unique identifier of the document, specified by *Index* .

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCview::GetDocumentIndex, IPDFXCview::GetDocumentsCount, Operations::GetDocumentID

2.1.1.2.9 GetDocumentIndex

Returns the unique identifier of the opened document which is specified by order index. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT GetDocumentIndex(
       LONG ID,
       LONG* Index
);
```

Parameters

ID

[in] **LONG** that specifies the unique identifier of the opened document. This argument can be 0 for the active document.

Index

[out] Pointer to a **LONG** that receives order index document, specified by *ID*.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCview::GetDocumentID, IPDFXCview::GetDocumentsCount, Operations::GetDocumentIndex

2.1.1.2.10 GetDocumentProperty

This method allows you to get property values for the specified document. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT GetDocumentProperty(
       LONG ID,
       BSTR Name,
       VARIANT* DataOut,
       LONG Flags
);
```

Parameters

ID
[in] **LONG** that specifies the unique identifier of the opened document. This argument can be 0 for the active document.

Name
[in] **BSTR** that specifies the full name of the document's property as string identifier. For more information, see Objects::Documents.

DataOut
[out] Pointer to a **VARIANT** structure that receives value of the property specified by *Name*.

Flags
[in] **LONG** that specifies optional flags. This argument can be 0.
Also you can pass the PXCVA_Flags::PXCVA_GetNamed to obtain *named value* (as a simple string) of the property specified by *Name*.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

Unlike the GetProperty, the name of document's property does not contain the document identification prefix (*"Documents[#4095]."* for example) because document is specified by *ID* argument already.
I.e., the call (in pseudocode):

```
GetProperty("Documents[#4095].Title", ...);
```

is equivalent to call:

```
GetDocumentProperty(4095, "Title", ...);
```

See Also

IPDFXCview::SetDocumentProperty, IPDFXCview::DoDocumentVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Objects::Documents

2.1.1.2.11 GetDocumentsCount

Returns the number of all opened documents. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT GetDocumentsCount(
       LONG* Count
);
```

Parameters

Count
[out] Pointer to a **LONG** that receives unique identifier of the active document.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCview::GetDocumentID, IPDFXCview::GetDocumentIndex,

Operations::GetDocumentsCount

2.1.1.2.12 OpenDocument

Open a document by the specified local file name, URL or by open file dialog. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT OpenDocument(
       BSTR SourceFileName,
       BSTR OpenPassword,
       LONG* ID,
       LONG Flags
);
```

Parameters

SourceFileName
[in] **BSTR** that specifies the local file name or URL. This argument can be NULL.

OpenPassword
[in] **BSTR** that determines the password to open the document specified by *SourceFileName*. This argument can be NULL.

ID
[out] Pointer to a **LONG** that receives the unique identifier of the opened document. This argument can be NULL.

Flags
[in] **LONG** that specifies optional flags. For more information, see PXCVA_Flags. This argument can be 0.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

To display standard open file dialog you should pass NULL to *SourceFileName*. If you want to deny all UI prompts, alerts (enter password dialog, error dialog, etc.) when calling this method you must specify the PXCVA_Flags::PXCVA_NoUI flag into *Flags*.

See Also

IPDFXCview::CloseDocument,
Operations::OpenDocument

2.1.1.2.13 PrintDocument

Print the opened document, with the parameters defined earlier by Objects::Print names section. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT PrintDocument(
       LONG ID,
       LONG Flags
);
```

Parameters

ID
[in] **LONG** that specifies unique identifier of the opened document. This argument can be 0 for the active document.

Flags
[in] **LONG** that specifies optional flags. For more information, see PXCVA_Flags. This argument can be 0.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

Call for print the opened document, specified by *ID*. If you want to skip the print dialog then you must specify the PXCVA_Flags::PXCVA_NoUI flag into *Flags*.

See Also

IPDFXCview::ExportDocument,
Objects::Print,
Operations::PrintDocument

2.1.1.2.14 SaveDocument

Save the opened document. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT SaveDocument(
       LONG ID,
       BSTR DestFileName,
       LONG SaveMode,
       LONG Flags,
);
```

Parameters

ID
[in] **LONG** that identifies the opened document. This argument can be 0 for the active document.

DestFileName
[in] **BSTR** that specifies destination file name. This argument can be NULL.

SaveMode
[in] **LONG** that specifies special save flags of document. For more information, see PXCVA_DocumentSaveFlags. This argument can be 0.

Flags
[in] **LONG** that specifies optional flags. For more information, see PXCVA_Flags. This argument can be 0.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

Call to save or copy the opened document, specified by *ID*. To display the standard save file dialog you should pass NULL to *DestFileName*. If you want to skip possible error dialogs then you must specify the PXCVA_Flags::PXCVA_NoUI flag in *Flags*.

See Also

IPDFXCview::OpenDocument, IPDFXCview::FlushDocument,
Objects::Documents::<Item>::Save,
Objects::Notifications::BeforeSaveDoc, Objects::Notifications::DocSaved,
Objects::Documents::UseStreamsDirectly

2.1.1.2.15 SetDocumentProperty

This method allows you to set property values for the specified document. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT SetDocumentProperty(
       LONG ID,
       BSTR Name,
       VARIANT DataIn,
       LONG Flags
);
```

Parameters

ID
[in] **LONG** that specifies the unique identifier of the opened document. This argument can be 0 for the active document.

Name
[in] **BSTR** that specifies the full name of the document's property as string identifier. For more information, see Objects::Documents.

DataIn
[in] **VARIANT** that specifies the new value of the document's property specified by *Name*.

Flags
[in] **LONG** that specifies optional flags. This argument can be 0.
Also you can pass the PXCVA_Flags::PXCVA_NoApply to instruct caching of passed value only.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

Unlike the SetProperty, the name of document's property does not contain the document identification prefix (*"Documents[#4095]."* for example) because the document is specified by *ID* argument already.
I.e., the call (in pseudocode):

```
SetProperty("Documents[#4095].Title", ...);
```

is equivalent to call:

```
SetDocumentProperty(4095, "Title", ...);
```

See Also

IPDFXCview::GetDocumentProperty, IPDFXCview::DoDocumentVerb,

IPDFXCview::GetProperty, IPDFXCview::SetProperty, Objects::Documents

2.1.1.3 Auxiliary Methods

Methods

Method	Description
GetDocumentFromName	This method is used to obtain unique identifier of document which is represented by input name string.
GetTextFromResult	This method is used to obtain the text description of returned error codes by other methods.
GetViewObjectFromName	This method is used to obtain type, identifier, proper-name of view object which is represented by input name string.

See Also

General Methods, Document Methods

2.1.1.3.1 GetDocumentFromName

This method is used to obtain the unique identifier of the open document which is represented by input name string. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT GetDocumentFromName(
       BSTR Name,
       LONG* ID,
       LONG* Eaten
);
```

Parameters

Name
[in] **BSTR** that specifies the name of document object. For more information, see Objects:: Documents.

ID
[out] Pointer to a **LONG** value that receives the unique identifier of detected document. If your application does not need this information, set it to NULL.

Eaten
[out] Pointer to a **LONG** value that receives the number of characters of the name that was parsed. If your application does not need this information, set it to NULL.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

Call this method to identify the object name which will be received by _IPDFXCviewEvents:: OnEvent notification. For example (in pseudocode):

```
GetDocumentFromName("Documents[#4095].Panes["Thumbnails"].Visible",
                    ID, Eaten);
// result:
ID == 4095;
Eaten == 17;
```

See Also

IPDFXCview::GetViewObjectFromName,
Operations::GetDocumentFromName

2.1.1.3.2 GetTextFromResult

This method is used to obtain the text description of returned error codes by other methods. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT GetTextFromResult(
       HRESULT Result,
       BSTR* Text
);
```

Parameters

Result
[in] **HRESULT** that specifies the result code which is returned by other methods.

Text
[out] Pointer to a **BSTR** that receives the descriptive text about the code specified by *Result*.

Return Value

Returns **S_OK** if successful, or an error value otherwise.

See Also

IPDFXCview::DoVerb, IPDFXCview::DoDocumentVerb,
Operations::GetTextFromResult

2.1.1.3.3 GetViewObjectFromName

This method is used to obtain the type, identifier, and proper name of a view object which is represented by the input name string. This method is a simplified variant of a special call to the DoVerb method.

Syntax

```
HRESULT GetViewObjectFromName(
       BSTR Name,
       LONG* Type,
       LONG* ID,
       BSTR* ProperName,
       LONG* Eaten
);
```

Parameters

Name
[in] **BSTR** that specifies the name of view object.
For more information, see Objects::View, Objects::Documents::<Item>.View.

Type
[out] Pointer to a **LONG** value that receives the type of detected view object. For more information, see PXCVA_ViewObjectTypes. If your application does not need this

information, set it to NULL.

ID

[out] Pointer to a **LONG** value that receives the unique identifier (if any) of the detected view object. If your application does not need this information, set it to NULL.

ProperName

[out] Pointer to a **BSTR** value that receives the proper name of detected view object. If your application does not need this information, set it to NULL.

Eaten

[out] Pointer to a **LONG** value that receives the number of characters of the name that was parsed. If your application does not need this information, set it to NULL.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

Call this method to recognize object name which is received through _IPDFXCviewEvents::OnEvent notification. For example (in pseudocode):

```
GetViewObjectFromName("View.Bars[\"Menu\"].Visible",
                      Type, ID, ProperName, Eaten);
// result:
Type == PXCVA_ViewObjectType::PXCVA_Bar;
ID == 33009;
ProperName == "Menu";
Eaten == 18;
```

See Also

IPDFXCview::GetDocumentFromName,
Operations::GetViewObjectFromName

2.1.1.4 Properties

Properties

Property Name	Description
Property	The common property accessor, parameterized by name.

See Also

General Methods, Document Methods, Auxiliary Methods

2.1.1.4.1 Property

Sets or gets the any property which specified by name.

Syntax

```
HRESULT get_Property(BSTR Name, LONG Flags, VARIANT* DataOut);
HRESULT put_Property(BSTR Name, LONG Flags, VARIANT DataIn);
```

Parameters

Name

[in] **BSTR** that specifies full name of the property as string identifier.
For more information, see Named Objects.

DataOut

[out] Pointer to a **VARIANT** structure that receives value of the property specified by *Name*.

`DataIn`

[in] **VARIANT** that specifies the new value of the property specified by *Name*.

Flags

[in] **LONG** that specifies optional flags. This argument can be 0.
Also you can pass the PXCVA_Flags::PXCVA_GetNamed to obtain the *named value* (as a simple string) of the property specified by *Name*.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.1.2 IPDFXCview2

This interface is the **default** interface of our ActiveX control (Microsoft ActiveX control). This interface is derived directly from IPDFXCview.

- Properties

Interface Identificator:

```
GUID(IID_IPDFXCview2): {53D68C77-20E5-455b-AA8E-2F071530FE67}
```

See Also

IPDFXCview

2.1.2.1 Properties

Properties

Property Name	Description
AllowAccelerators	Allows or denies all control's accelerators.
LockedView	Allows or denies all UI-customization.
ReadOnly	Allows or denies all modification operations, for all documents.
SettingsURL	Sets or gets the *URL* to the *settings file* for control customizing.
Src	Sets/gets the *source-URL* of the document for open.

See Also

IPDFXCview2

2.1.2.1.1 AllowAccelerators

Allows or denies all control's accelerators.

Syntax

```
HRESULT get_AllowAccelerators(VARIANT_BOOL* ValueOut);
HRESULT put_AllowAccelerators(VARIANT_BOOL ValueIn);
```

Parameters

ValueOut
[out] Pointer to a **VARIANT_BOOL** that receives flag for accelerators usage.
ValueIn
[in] **BSTR** that specifies flag for accelerators usage.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

Objects::General.AllowAllAccelerators

2.1.2.1.2 LockedView

Allows or denies all UI-customization.

Syntax

```
HRESULT get_LockedView(VARIANT_BOOL* ValueOut);
HRESULT put_LockedView(VARIANT_BOOL ValueIn);
```

Parameters

ValueOut
[out] Pointer to a **VARIANT_BOOL** that receives flag for view locking.
ValueIn
[in] **BSTR** that specifies flag for view locking.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

Objects::View.Locked

2.1.2.1.3 ReadOnly

Allows or denies all modification operations, for all documents.

Syntax

```
HRESULT get_ReadOnly(VARIANT_BOOL* ValueOut);
HRESULT put_ReadOnly(VARIANT_BOOL ValueIn);
```

Parameters

ValueOut
[out] Pointer to a **VARIANT_BOOL** that receives *read-only* flag.
ValueIn
[in] **BSTR** that specifies *read-only* flag.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

Objects::General.DenyAllModifyOperations

2.1.2.1.4 SettingsURL

Sets or gets the URL to the settings.dat file with all stored settings for the control. Simply specify the correct path location to the file and the control will load and use these stored settings.

Example:

http://www.mysite.com/settings.dat
file:///C:/settings.dat
C:\settings.dat

Syntax

```
HRESULT get_SettingsURL(BSTR* ValueOut);
HRESULT put_SettingsURL(BSTR ValueIn);
```

Parameters

ValueOut
[out] Pointer to a **BSTR** that receives source-URL of the file with control's settings.
ValueIn
[in] **BSTR** that specifies source-URL of the file with control's settings.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCview::LoadSettings,
Operations::LoadSettings

2.1.2.1.5 Src

Sets or gets the *source-URL* of a document. Simply specify correct path to a document and the control will open the required file.
The source-URL may contain some specific 'open' actions(see PDF Open Parameters reference), for example:

http://www.mysite.com/test.pdf#page=2&zoom=150
file:///C:/test.pdf
C:\test.pdf

Syntax

```
HRESULT get_Src(BSTR* ValueOut);
```

```
HRESULT put_Src(BSTR ValueIn);
```

Parameters

ValueOut
[out] Pointer to a **BSTR** that receives source-URL of the document.
ValueIn
[in] **BSTR** that specifies source-URL of the document.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCview::OpenDocument,
Operations::OpenDocument

2.1.3 _IPDFXCviewEvents

This interface designates an event sink interface that an application must implement in order to receive event notifications from a control.

Method	Description
OnEvent	Fires when property of object is changed, notifies about some prompts, actions.

Interface Identificator:

```
GUID(DIID__IPDFXCviewEvents): {C5EA83BB-986F-4F32-AD38-E47C3970A357}
```

See Also

PXCVA_EventTypes, IPDFXCview::GetViewObjectFromName, IPDFXCview::GetDocumentFromName,
Named Objects, Objects::Prompts, Objects::Notifications

2.1.3.1 OnEvent

Fires when a control property is changed, or notifies for some prompts and other object changes.

Syntax

```
HRESULT OnEvent(
       LONG Type,
       BSTR Name,
       VARIANT DataIn,
       VARIANT* DataOut,
       LONG Flags
);
```

Parameters

Type

[in] **LONG** that specifies the event type. For more information, see PXCVA_EventTypes.

Name

[in] **BSTR** that specifies the unique event name. Usually it contains full name of object-initiator or other special name of the event (dependent on *Type*). See also Named Objects, Objects::Notifications.

DataIn

[in] **VARIANT** that specifies the necessary input data dependent with *Type* and *Name* arguments. This argument may be empty.

DataOut

[out] Pointer to a **VARIANT** structure that receives output data dependent with *Type* and *Name* arguments. This argument can be NULL.

Flags

[in] **LONG** that specifies optional flags. For more information, see PXCVA_Flags. This argument can be 0.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCview::GetViewObjectFromName, IPDFXCview::GetDocumentFromName, Named Objects, Objects::Prompts, Objects::Notifications

2.1.4 IPDFXCargs

This interface represents an array of data-objects. Use it for simplification creation/reading/changing of data-object which contains array of data(**SAFEARRAY**): pass filled *args-object* as input-data to any other methods through **DataIn**, obtain new *args-object* as output-data through **DataOut**. This interface is derived directly from **IDispatch**.

- Methods
- Properties

Remarks

To obtain a arguments object, you should call main method IPDFXCview::DoVerb with special named operationOperations::.ARGS. Or you may obtain new *args-object* as output data from any method like as IPDFXCview::DoVerb if you have specified PXCVA_Flags::PXCVA_OutArgs flag into **Flags**-argument before.

For example:

```
C++,ATL
CComPtr<IPDFXCview> spView;
CComPtr<IPDFXCargs> spArgs;
...
CComBSTR objName(L"");
CComBSTR opName(L".ARGS");
CComVariant dataIn, dataOut;
spView->DoVerb(objName, opName, dataIn, &dataOut, 0);
spArgs = static_cast<IPDFXCargs*>(dataOut.punkVal);
...
spArgs->Init(2);
spArgs->put_Item(0, "http://www.mysite.com/test.pdf#page=2&zoom=150");
spArgs->put_Item(1, "mypassword");
dataIn = spArgs;
opName = L"OpenDocument";
spView->DoVerb(objName, opName, dataIn, &dataOut, 0);
```

```
C#
PDFXCviewAxLib.IPDFXCargs args;
object dataOut, dataIn;
dataIn = null;
...
axCoPDFXCview1.DoVerb("", ".ARGS", dataIn, out dataOut, 0);
args = (PDFXCviewAxLib.IPDFXCargs)dataOut;
...
args.Init(2);
args[0] = "http://www.mysite.com/test.pdf#page=2&zoom=150";
args[1] = "mypassword";
axCoPDFXCview1.DoVerb("", "OpenDocument", args, out dataOut);

VB.NET
Dim args As PDFXCviewAxLib.IPDFXCargs
Dim dataOut, dataIn As Object
dataIn = Nothing
dataOut = Nothing
...
AxCoPDFXCview1.DoVerb("", ".ARGS", dataIn, dataOut, 0)
args = CType(dataOut, PDFXCviewAxLib.IPDFXCargs)
...
args.Init(2)
args(0) = "http://www.mysite.com/test.pdf#page=2&zoom=150"
args(1) = "mypassword"
AxCoPDFXCview1.DoVerb("", "OpenDocument", PDFXCargs, dataOut)

VB6
Dim args As PDFXCviewAxLibCtl.IPDFXCargs
Dim dataIn, dataOut As Variant
...
Call CoPDFXCview1.DoVerb("", ".ARGS", dataIn, dataOut, 0)
Set args = dataOut
...
Call args.Init(2)
args(0) = "http://www.mysite.com/test.pdf#page=2&zoom=150"
args(1) = "mypassword"
Call CoPDFXCview1.DoVerb("", "OpenDocument", args, dataOut)

DELPHI
var
  args:IPDFXCargs;
  dataIn, dataOut:OleVariant;
...
dataIn := '';
CoPDFXCview1.DoVerb('', '.ARGS', dataIn, dataOut, 0);
args:= IDispatch(dataOut) as IPDFXCargs;
...
args.Init(2);
args[0]:='http://www.mysite.com/test.pdf#page=2&zoom=150';
args[1]:='mypassword';
CoPDFXCview1.DoVerb('', 'OpenDocument', args, dataOut, 0);
```

Interface Identificator:

```
GUID(IID_IPDFXCargs): {C7F70A09-F3F8-4344-A688-6BC747A694B6}
```

See Also

IPDFXCview, IPDFXCsmartp,
Operations::.ARGS, PXCVA_Flags::PXCVA_OutArgs

2.1.4.1 Methods

Methods

Method	Description
Add	Appends a new item to the collection.
Clear	Removes all items from the collection.
Init	Creates new collection with the specified size.
Remove	Removes the item at the specified index from the collection.

See Also

IPDFXCview, IPDFXCsmartp

2.1.4.1.1 Add

Appends a new item to the collection.

Syntax

```
HRESULT Add(VARIANT ValueIn);
```

Parameters

ValueIn
[in] **VARIANT** that specifies the value of the new collection item.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of received error code you can use GetTextFromResult.

2.1.4.1.2 Clear

Removes all items from the collection.

Syntax

```
HRESULT Clear();
```

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of received error code you can use GetTextFromResult.

2.1.4.1.3 Init

Creates new collection with the specified size.

Syntax

```
HRESULT Init(LONG CountIn);
```

Parameters

CountIn
[in] **LONG** that specifies the count of items in the new collection.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of received error code you can use GetTextFromResult.

2.1.4.1.4 Remove

Removes the item at the specified index from the collection.

Syntax

```
HRESULT Remove(LONG Index);
```

Parameters

Index
[in] **LONG** that specifies the index (0-based) of the item to be removed.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of received error code you can use GetTextFromResult.

2.1.4.2 Properties

Properties

Property Name	Description
_NewEnum	Returns an **IEnumVARIANT** interface that can be used to enumerate the collection.
Count	Indicates the number of items contained in the collection.
Data	Gets/sets collection data, i.e. the variant-object(**VARIANT**) which can contain a **SAFEARRAY**.
Item	Gets/sets value of an collection item(variant-object) by index.

See Also

IPDFXCview, IPDFXCsmartp

2.1.4.2.1 _NewEnum

Returns an **IEnumVARIANT** interface that can be used to enumerate the collection.

Syntax

```
HRESULT get__NewEnum(IUnknown** ppUnkOut);
```

Parameters

ppUnkOut
[out] Double pointer to a **IUnknown** that receives an **IEnumVARIANT** interface which can be used to enumerate the collection.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

2.1.4.2.2 Count

Indicates the number of elements contained in the collection.

Syntax

```
HRESULT get_Count(LONG* CountOut);
```

Parameters

CountOut
[out] Pointer to a **LONG** that receives the number of elements contained in the collection.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

2.1.4.2.3 Data

Provides get/set collection data, i.e. the variant-object(**VARIANT**) which can contain a **SAFEARRAY**.

Syntax

```
HRESULT get_Data(VARIANT* ValueOut);
HRESULT put_Data(VARIANT ValueIn);
```

Parameters

ValueOut
[out] Pointer to a **VARIANT** that receives collection data.
ValueIn
[in] **BSTR** that specifies a new collection data.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

2.1.4.2.4 Item

Provides get/set value of an collection item(variant-object) by index.

Syntax

```
HRESULT get_Item(LONG Index, VARIANT* ValueOut);
HRESULT put_Item(LONG Index, VARIANT ValueIn);
```

Parameters

Index
[in] **LONG** that specifies a position(0-based) of collection item to get/set.
ValueOut
[out] Pointer to a **VARIANT** that receives collection item value.
ValueIn
[in] **BSTR** that specifies a new collection item value.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

2.1.5 IPDFXCsmartp

This interface represents special object (Smart Point) for simplification and access acceleration to any objects or properties which are described in the section Named Objects.

- Methods
- Properties

Remarks

To obtain a primary smart point object, you should call main method IPDFXCview::DoVerb with special named operation Operations::.SP.

For example:

```
C++,ATL
CString str;
CComPtr<IPDFXCview> spView;
CComPtr<IPDFXCsmartp> spDoc;
...
str.Format(L"Documents[#%d]", iDocID);
CComBSTR objName((LPCWSTR)str);
CComBSTR opName(L".SP");
CComVariant dataIn, dataOut;
spView->DoVerb(objName, opName, dataIn, &dataOut, 0);
spDoc = static_cast<IPDFXCsmartp*>(dataOut.punkVal);
...
objName = L"Pages.Count";
spDoc->GetProperty(objName, &dataOut, 0); // gets pages count for specifie

C#
PDFXCviewAxLib.IPDFXCsmartp spDoc;
object dataIn = null;
object dataOut = null;
axCoPDFXCview1.DoVerb("Documents[#" + iDocID.ToString() + "]", ".SP", data
spDoc = (PDFXCviewAxLib.IPDFXCsmartp)dataOut;
...
spDoc.GetProperty("Pages.Count", out dataOut, 0); // gets pages count for

VB.NET
Dim spDoc As PDFXCviewAxLib.IPDFXCsmartp
Dim dataIn As Object = Nothing
Dim dataOut As Object = Nothing
AxCoPDFXCview1.DoVerb("Documents[#" + iDocID.ToString() + "]", ".SP", data
spDoc = dataOut
...
spDoc.GetProperty("Pages.Count", dataOut, 0); 'gets pages count for specif

VB6
Dim dataIn As Variant
Dim dataOut As Variant
Dim spDoc As IPDFXCsmartp
dataIn = 0
Dim objName As String
objName = "Documents[#" + LTrim$(Str(iDocID)) + "]"
Call CoPDFXCview1.DoVerb(objName, ".SP", dataIn, dataOut, 0)
Set spDoc = dataOut
...
Call spDoc.GetProperty("Pages.Count", dataOut, 0) 'gets pages count for sp
```

```
DELPHI
var
  spDoc: IPDFXCsmartp;
  dataIn, dataOut:OleVariant;
  iDocID:integer;
...
CoPDFXCview1.DoVerb('Documents[#'+inttostr(iDocID)+']', '.SP', dataIn, dat
spDoc := IDispatch(dataOut) as IPDFXCsmartp;
...
spDoc.GetProperty('Pages.Count', dataOut, 0); // gets pages count for spec
```

Interface Identificator:

```
GUID(IID_IPDFXCsmartp): {256342AE-1477-4722-BA9D-4A2AE6984494}
```

See Also

IPDFXCview::DoVerb, Operations::.SP

2.1.5.1 Methods

Methods

Method	Description
DoVerb	Provides any supported functionality for the smart point object.
GetItemPointByID	If this smart point represents an object which contains an array of items, you can create a new smart point object for the item which is specified by a unique ID (if it is supported) and obtain interface pointer to it.
GetItemPointByIndex	If this smart point represents an object which contains an array of items, you can create a new smart point object for the item which is specified by order index and obtain interface pointer to it.
GetItemPointByNIndex	If this smart point represents an object which contains an array of items, you can create a new smart point object for the item which is specified by a unique name (if it is supported) and obtain interface pointer to it.
GetPointName	Retrieves full name of this smart point.
GetProperty	Retrieves value of specified sub-property by relative name.
GetSubPoint	Creates a new smart point object by specified relative name and retrieves interface pointer to him.
SetProperty	Sets value of specified sub-property by relative name.

See Also

IPDFXCview, IPDFXCargs

2.1.5.1.1 DoVerb

This is a shortcut to the **main method** IPDFXCview::DoVerb in the library, but **ObjectName** (if not *NULL*) argument should mean relative name of any sub-object or sub-property.

The syntax of this method is identical to main IPDFXCview::DoVerb.

2.1.5.1.2 GetItemPointByID

If this smart point represents an object which contains an array of items, you can create a new smart point object for the item which is specified by a unique ID (if it is supported) and obtain interface pointer to it.

Syntax

```
HRESULT GetItemPointByID(
       LONG UniqueID,
       IPDFXCsmartp** ItemPointOut
);
```

Parameters

UniqueID
[in] **LONG** that specifies the unique ID (if it is supported) of item.
ItemPointOut
[out] Pointer to a **IPDFXCsmartp** interface that receives new smart point object which depends on an item specified by *UniqueID*.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

This method requires a special numeric property in item structure with name "**ID**" which contains a unique identifier of the item in current array.
Examples of the item structure that contains property "**ID**":
Objects::Documents::<Item>,
Objects::Bars::<Item>
Objects::Commands::<Item>

See Also

IPDFXCsmartp

2.1.5.1.3 GetItemPointByIndex

If this smart point represents an object which contains an array of items, you can create a new smart point object for the item which is specified by order index and obtain interface pointer to it.

Syntax

```
HRESULT GetItemPointByIndex(
       LONG Index,
       IPDFXCsmartp** ItemPointOut
);
```

Parameters

Index
[in] **LONG** that specifies the order index of item
ItemPointOut
[out] Pointer to a **IPDFXCsmartp** interface that receives a new smart point object which depends on an item specified by *Index*.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCsmartp

2.1.5.1.4 GetItemPointByNIndex

If this smart point represents an object which contains an array of items, you can create a new smart point object for the item which is specified by a unique name (if it is supported) and obtain interface pointer to it.

Syntax

```
HRESULT GetItemPointByNIndex(
       BSTR NamedIndex,
       IPDFXCsmartp** ItemPointOut
);
```

Parameters

NamedIndex
[in] **BSTR** that specifies the unique name (if it is supported) of the item.

ItemPointOut
[out] Pointer to a **IPDFXCsmartp** interface that receives new smart point object which depends on an item specified by *NamedIndex*.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

Remarks

This method requires a special string property in item structure with name "**Name**" which contains a unique identifier of the item in current array.
Examples of the item structure that contains property "**Name**":
Objects::Documents::<Item>,
Objects::Bars::<Item>

See Also

IPDFXCsmartp

2.1.5.1.5 GetPointName

Retrieves full name of this smart point.

Syntax

```
HRESULT GetPointName(
       BSTR* PointNameOut
);
```

Parameters

PointNameOut
[out] Pointer to a **BSTR** that receives full name of this smart point.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCsmartp

2.1.5.1.6 GetProperty

Retrieves value of specified sub-property by relative name. The syntax of this method is identical to main IPDFXCview::GetProperty.

2.1.5.1.7 GetSubPoint

Creates a new smart point object by specified relative name and retrieves interface pointer to him.

Syntax

```
HRESULT GetSubPoint(
       BSTR SubName,
       IPDFXCsmartp** SubPointOut
);
```

Parameters

SubName
[in] **BSTR** that specifies the relative name of any sub-object.
SubPointOut
[out] Pointer to a **IPDFXCsmartp** interface that receives a new smart point object which depends on an object specified by *SubName*.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCsmartp

2.1.5.1.8 SetProperty

Sets value of specified sub-property by relative name. The syntax of this method is identical to main IPDFXCview::SetProperty.

2.1.5.2 Properties

Properties

Property Name	Description

Property	The common property accessor, parameterized by relative name. Sets/Gets value of specified sub-property by relative name.

See Also

IPDFXCview, IPDFXCargs

2.1.5.2.1 Property

The common property accessor, parameterized by relative name. Sets/Gets value of specified sub-property by relative name.

The syntax of this method is identical to main IPDFXCview::Property.

2.2 Named Items

In order to assist the developer in developing applications using our **PDF-XChange Viewer ActiveX**, "shortcuts" have been created for some operations, objects and values, collectively called "Named Items."

This section consists of the following sections:

- Simple Operations
- Objects
- Values
- Simple Notifications

2.2.1 Simple Operations

Some of our methods require a *special named operation* as input string-argument (**OperationName** argument in IPDFXCview::DoVerb, IPDFXCview::DoDocumentVerb).

Common Object-Dependent Operations

These operations can be used to obtain or set values of properties:

Name	Description
Get	Obtains current values from properties.
GetNamed	Obtains current named values from properties. The same effect with PXCVA_Flags::PXCVA_GetNamed flag usage.
GetN	The same as *GetNamed* operation.
Set	Sets new values to properties.
SetNoApply	*Caches* new values to properties instead of applying them immediately. The same effect with PXCVA_Flags::PXCVA_NoApply flag usage.
SetNA	The same as *SetNoApply* operation.

These operations can be used to obtain some new auxiliary objects.

Name	Description
.ARGS	This operation can be used to obtain a new object with IPDFXCargs interface.
.SP	This operation can be used to obtain a primary smart point object (object with IPDFXCsmartp interface) by specified object name.

General Operations

Note: for all operations below the input argument, **ObjectName** (see IPDFXCview::DoVerb) is not needed, and should be set to NULL. For these operations the IPDFXCview::DoDocumentVerb method cannot be used.

Name	Description
ActivateDocument	Activates an open document.
ApplyAllCachedChanges	Applies all cached settings that were passed with the PXCVA_Flags::PXCVA_NoApply flag.
ClearRecentsList	This operation can be used to clear list of recent documents.
CloseAllDocuments	Closes all open documents.
CloseDocument	Closes an open document.
CreateNewBlankDocument	Creates new PDF document.
DeleteDocumentPages	Delete pages from PDF document using parameters previously defined by Objects::Operations.DeletePages.
DiscardAllCachedChanges	Discards all cached settings that were passed with the PXCVA_Flags::PXCVA_NoApply flag.
ExecuteCommand	Performs specified UI command.
ExportDocument	Exports a document to an image format with the previously defined parameters in Objects::Export.
ExtractDocumentPages	Extracts pages from PDF document using parameters previously defined by Objects::Operations.ExtractPages.
FlushDocument	Flushes all of the user's cached changes for a document.
GetActiveDocument	Returns the unique identifier (ID) of the active document.
GetDocumentFromName	Obtains the unique identifier of the named document.
GetDocumentID	Returns the document unique identifier (ID) for the specified index.
GetDocumentIndex	Returns the document order index either by the unique identifier (ID) or by the full file name.
GetDocumentsCount	Returns the number of all opened documents.
GetTextFromResult	Obtains the text description of error codes returned by other operations.
GetViewObjectFromName	Obtains type, identifier, and proper name of the view object.
HasVisibleBars	Tests visibility of UI-bars (menubar, toolbars, status bar, etc.).
InsertEmptyDocumentPages	Inserts empty pages into document.
InsertDocumentPages	Inserts pages from source document.
IsPDF	Tests file or stream object for PDF format.
MsgToCommand	Translates the standard windows-message to an existing UI-command.
MsgToCommandAndExec	Translates the standard windows-message to an existing UI-command and executes it.
NewDocumentFromImages	Creates new PDF document from the images.
NewDocumentFromText	Creates new PDF document from the plain text.
NewDocumentFromRTF	Creates new PDF document from the formatted text (RTF).
LoadSettings	Loads all saved control settings.
OpenDocument	Opens a document.
PrintDocument	Prints a document with the parameters defined earlier by

	Objects::Print.
RemoveRecentItem	This operation can be used to remove existing item from list of recent documents.
RotateDocumentPages	Rotates document pages.
RunJavaScript	Executes the supplied Java Script.
SaveDocument	Saves a document.
SaveSettings	Saves all control settings to external storage.
ShowStampsCollection	Shows/hides the specified stamps collection.
SummarizeDocumentAnnots	Summarizes all comments from a document to a new document (file) using parameters previously defined by Objects::Operations.SummarizeAnnots.

Remarks

Names of all operations are *case-independent,* i.e. the two names *"GetNamed"* and *"getNamed"* are considered identical for all operation-input methods.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, IPDFXCview::DoDocumentVerb

2.2.1.1 .ARGS

This operation can be used to obtain a new object with IPDFXCargs interface.

Name

```
.ARGS
```

Arguments

```
None
```

Return Value

Returns **S_OK** or an error value otherwise. For obtaining text description of received error code you can use IPDFXCview::GetTextFromResult.

If successful it returns in **DataOut**, an address of a pointer to **IDispatch** interface that receives the new arguments object.

See Also

IPDFXCview::DoVerb, PXCVA_Flags::PXCVA_OutArgs

2.2.1.2 .SP

This operation can be used to obtain a primary smart point object by specified object name.

Name

```
.SP
```

Arguments

#	Req.	Type	Description
1	Yes	BSTR	The full name of any object which is described in section Objects. For more information, see IPDFXCsmartp interface.

Return Value

Returns **S_OK** or an error value otherwise. For obtaining text description of received error code you can use IPDFXCview::GetTextFromResult.

If successful it returns in **DataOut**, an address of a pointer to **IDispatch** interface that receives the new smart point object.

See Also

IPDFXCview::DoVerb

2.2.1.3 ActivateDocument

Activates an open document.

Name

```
ActivateDocument
```

Arguments

#	Req.	Type	Description
1	Yes	LONG /BSTR	The unique identifier of the document or full file name of the document.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb, IPDFXCview::ActivateDocument

2.2.1.4 ApplyAllCachedChanges

This operation can be used to apply all cached settings that were passed with the PXCVA_Flags:: PXCVA_NoApply flag.

Name

```
ApplyAllCachedChanges
```

Arguments

Nothing

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb, IPDFXCview::ApplyAllCachedChanges

2.2.1.5 ClearRecentsList

This operation can be used to clear list of recent documents.

Name

```
ClearRecentsList
```

Arguments

Nothing

Return Value

Returns **S_OK** or an error value otherwise. For obtaining text description of received error code you can use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb

2.2.1.6 CloseAllDocuments

This operation can be used to close all opened documents.

Name

```
CloseAllDocuments
```

Arguments

Nothing

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb, IPDFXCview::CloseAllDocuments

2.2.1.7 CloseDocument

Closes an opened document.

Name

```
CloseDocument
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The unique identifier of the document or full file name of the document.

		/BSTR	

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb, IPDFXCview::CloseDocument

2.2.1.8 CreateNewBlankDocument

Creates new PDF document using parameters previously defined by Objects::Operations.NewDocument.FromBlank.

Name

```
CreateNewBlankDocument
```

or short:

```
NewBlankDocument
```

Arguments

Nothing

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

If successful it returns the unique identifier (ID) of the created blank document in **DataOut.**

See Also

IPDFXCview::DoVerb, Objects::Operations.NewDocument.FromBlank

2.2.1.9 DeleteDocumentPages

Delete pages from PDF document using parameters previously defined by Objects::Operations.DeletePages.

Name

```
DeleteDocumentPages
```

Arguments

#	Req.	Type	Description
1	Yes	LONG /BSTR	The unique identifier of the document or full file name of the document.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a

received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb,
Objects::Operations.DeletePages
Objects::Documents::<Item>::DeletePages

2.2.1.10 DiscardAllCachedChanges

Discards all cached settings that were passed with the PXCVA_Flags::PXCVA_NoApply flag.

Name

```
DiscardAllCachedChanges
```

Arguments

Nothing

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb, IPDFXCview::DiscardAllCachedChanges

2.2.1.11 ExecuteCommand

This operation can be used to perform a UI-command.

Name

```
ExecuteCommand
```

Arguments

#	Req.	Type	Description
1	Yes	LONG /BSTR	The unique identifier or special unique name of the UI-command. For more information, see Objects::Commands.

Return Value

Returns **S_OK** or an error value otherwise. For obtaining text description of received error code you can use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb,
Objects::Commands

2.2.1.12 ExportDocument

Exports a document to an image file(s) with the format parameters previously defined by Objects::Export .

Name

```
ExportDocument
```

Arguments

#	Req.	Type	Description
1	Yes	LONG /BSTR	The unique identifier of the document or full file name of the document.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb, IPDFXCview::ExportDocument

2.2.1.13 ExtractDocumentPages

Extracts pages from PDF document using parameters previously defined by Objects::Operations.ExtractPages.

Name

```
ExtractDocumentPages
```

Arguments

#	Req.	Type	Description
1	Yes	LONG /BSTR	The unique identifier of the document or full file name of the document.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb,
Objects::Operations.ExtractPages
Objects::Documents::<Item>::ExtractPages

2.2.1.14 FlushDocument

Flushes all of the user's cached changes for a document, stops editing and updates document's structure. Can be used before saving of the document.

Name

```
FlushDocument
```

Arguments

Nothing

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb, IPDFXCview::FlushDocument

2.2.1.15 GetActiveDocument

Returns the unique identifier of the active document.

Name

```
GetActiveDocument
```

Arguments

Nothing

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

If successful then returns the unique identifier of the active document in **DataOut.**

See Also

IPDFXCview::DoVerb, IPDFXCview::GetActiveDocument

2.2.1.16 GetDocumentFromName

Obtains unique identifier of document.

Name

```
GetDocumentFromName
```

Arguments

#	Req.	Type	Description
1	Yes	BSTR	Specifies the name of document object. For more information, see Objects::Documents.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.
If successful, **DataOut** contains:
1. The unique identifier of detected document.
2. The number of characters of the name that was parsed.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetDocumentFromName

2.2.1.17 GetDocumentID

Returns the document unique identifier by order index.

Name

```
GetDocumentID
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The index of the document.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

If successful the unique identifier (ID) of the indexed document will be returned in **DataOut.**

See Also

IPDFXCview::DoVerb, IPDFXCview::GetDocumentID

2.2.1.18 GetDocumentIndex

Returns the document order index by unique identifier or by full file name.

Name

```
GetDocumentIndex
```

Arguments

#	Req.	Type	Description
1	Yes	LONG /BSTR	The unique identifier of the document or full file name of the document.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

If successful then the index of the opened document will be returned in **DataOut.**

See Also

IPDFXCview::DoVerb, IPDFXCview::GetDocumentID

2.2.1.19 GetDocumentsCount

Returns the number of all opened documents.

Name

```
GetDocumentsCount
```

Arguments

Nothing

Return Value

Returns **S_OK** on success, on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

If successful then puts the number of all opened documents into the **DataOut.**

See Also

IPDFXCview::DoVerb, IPDFXCview::GetDocumentsCount

2.2.1.20 GetTextFromResult

Obtains the text description of returned error codes by other operations.

Name

```
GetTextFromResult
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	Specifies the result code which is returned by other operations.

Return Value

Returns **S_OK** on success, or an error value otherwise.
If successful, then **DataOut** contains the description text for the Result code..

See Also

IPDFXCview::DoVerb, IPDFXCview::GetTextFromResult

2.2.1.21 GetViewObjectFromName

Obtains type, identifier, proper-name of view object.

Name

```
GetViewObjectFromName
```

Arguments

#	Req.	Type	Description
1	Yes	BSTR	Specifies the name of view object. For more information, see Objects::View, Objects::Documents::<Item>.View.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.
If successful, **DataOut** contains:
1. The type of detected view object. For more information, see PXCVA_ViewObjectTypes.

2. The unique identifier of detected view object.
3. The proper name of detected view object.
4. The number of characters of the name that was parsed.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetViewObjectFromName

2.2.1.22 HasVisibleBars

Tests visibility of UI-bars (menubar, toolbars, status bar, etc.).

Name

```
HasVisibleBars
```

Arguments

Nothing

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

If successful then it returns in **DataOut** the non-zero number when any UI-bar is visible or zero otherwise.

See Also

IPDFXCview::DoVerb

2.2.1.23 InsertDocumentPages

Inserts pages from source document into current document using parameters previously defined by Objects::Operations.InsertPages.

Name

```
InsertDocumentPages
```

Arguments

#	Req.	Type	Description
1	Yes	LONG /BSTR	The unique identifier of the document or full file name of the document.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb,
Objects::Operations.InsertPages,
Objects::Documents::<Item>::InsertPages

2.2.1.24 InsertEmptyDocumentPages

Inserts empty pages to PDF document using parameters previously defined by Objects::Operations.InsertEmptyPages.

Name

```
InsertEmptyDocumentPages
```

Arguments

#	Req.	Type	Description
1	Yes	LONG /BSTR	The unique identifier of the document or full file name of the document.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb,
Objects::Operations.InsertEmptyPages,
Objects::Documents::<Item>::InsertEmptyPages

2.2.1.25 IsPDF

Tests file or stream object for PDF format.

Name

```
IsPDF
```

Arguments

#	Req.	Type	Description
1	Yes	BSTR/ IStrea m	Source file name, or valid URL, or Stream Object. Some examples: http://www.mysite.com/test.pdf file:///C:/test.pdf C:\test.pdf

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

If successful it returns in **DataOut** the non-zero number when the input source is PDF-document or zero otherwise.

Remarks

For example (in pseudocode):

```
...
DoVerb(NULL, "IsPDF", DataIn("C:\Test.pdf"), DataOut, 0);
// test result in DataOut
```

. . .

See Also

IPDFXCview::DoVerb, IPDFXCview::OpenDocument, Objects::Documents::<Item>.PDFA

2.2.1.26 LoadSettings

Loads all control settings from an external storage.

Name

```
LoadSettings
```

Arguments

#	Req.	Type	Description
1	Yes	BSTR/IStream	Specifies a valid source storage: source file name or pointer to a **IStream** interface.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb, IPDFXCview::LoadSettings

2.2.1.27 MsgToCommand

Translates the standard windows-message (WM_KEYDOWN, WM_SYSKEYDOWN) to an existing UI-command.

Name

```
MsgToCommand
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The windows-message identifier.
2	Yes	LONG	The message **wParam** parameter.
3	No	LONG	The message **lParam** parameter.
4	No	LONG	Alternate modifier flags set. Can contains one or more standard flags: **FCONTROL**(8), **FSHIFT**(4), **FALT**(16). If not specified then control will obtain these flags from system automatically.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

If successful it returns in **DataOut** the unique identifier (ID) of the existing UI-command or zero

otherwise.

Remarks

For example (in pseudocode):

```
...
// translates the Ctrl+O key combination
DoVerb(NULL, "MsgToCommand",
       DataIn(WM_KEYDOWN, 'O', 0, FCONTROL), DataOut, 0);
// DataOut == 57601; // The "Open" UI-command
...
```

See Also

IPDFXCview::DoVerb

2.2.1.28 MsgToCommandAndExec

Translates the standard windows-message (WM_KEYDOWN, WM_SYSKEYDOWN) to an existing UI-command and, if successful, executes it.
This operation is similar to the Operations::MsgToCommand.

Name

```
MsgToCommandAndExec
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The windows-message identifier.
2	Yes	LONG	The message **wParam** parameter.
3	No	LONG	The message **lParam** parameter.
4	No	LONG	Alternate modifier flags set. Can contains one or more standard flags: **FCONTROL**(8), **FSHIFT**(4), **FALT**(16). If not specified then control will obtain these flags from system automatically.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

If successful it returns in **DataOut** the non-zero or zero otherwise.

Remarks

For example (in pseudocode):

```
...
// Translates "Ctr+O" to the "Open" UI-command and executes it
DoVerb(NULL, "MsgToCommandAndExec",
       DataIn(WM_KEYDOWN, 'O', 0, FCONTROL), DataOut, 0);
...
```

See Also

IPDFXCview::DoVerb

2.2.1.29 NewBlankDocument

See the Operations::CreateNewBlankDocument.

2.2.1.30 NewDocumentFromImages

Creates new PDF document from the images using parameters previously defined by Objects::Operations.NewDocument.FromImages.

Name

```
NewDocumentFromImages
```

Arguments

#	Req.	Type	Description
1	Yes	BSTR	Image file name.
2	No	BSTR	Next image file name.
...			

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

If successful it returns the unique identifier (ID) of the created document in **DataOut.**

Remarks

You may use the object with IPDFXCargs interface to create the image files list as input data. For example (in pseudocode):

```
...
// create empty IPDFXCargs object
IPDFXCargs args;
DoVerb(NULL, ".ARGS", dataIn, dataOut, 0);
args = (IPDFXCargs)dataOut;
dataOut = NULL;
...
// create input files list
args.Add("C:\image1.jpg");
args.Add("C:\image2.png");
args.Add("C:\image3.tiff");
...
// create new document from the images
DoVerb(NULL, "NewDocumentFromImages", dataIn, dataOut, 0);
newDocID = dataOut;
...
```

See Also

IPDFXCview::DoVerb, Objects::Operations.NewDocument.FromImages

2.2.1.31 NewDocumentFromRTF

Creates new PDF document from the formatted text (RTF).

Name

```
NewDocumentFromRTF
```

Arguments

#	Req.	Type	Description
1	Yes	BSTR	RTF file name.
2	No	BSTR	Next RTF file name.
...			

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

If successful it returns the unique identifier (ID) of the created document in **DataOut.**

Remarks

You may use the object with IPDFXCargs interface to create the text files list as input data. For example (in pseudocode):

```
...
// create empty IPDFXCargs object
IPDFXCargs args;
DoVerb(NULL, ".ARGS", dataIn, dataOut, 0);
args = (IPDFXCargs)dataOut;
dataOut = NULL;
...
// create input files list
args.Add("C:\Test1.rtf");
args.Add("C:\Test2.rtf");
args.Add("C:\Test3.rtf");
...
// create new document from the formatted text files
DoVerb(NULL, "NewDocumentFromRTF", dataIn, dataOut, 0);
newDocID = dataOut;
...
```

See Also

IPDFXCview::DoVerb

2.2.1.32 NewDocumentFromText

Creates new PDF document from the plain text using parameters previously defined by Objects::Operations.NewDocument.FromText.

Name

```
NewDocumentFromText
```

Arguments

#	Req.	Type	Description
1	Yes	BSTR	Text file name.
2	No	BSTR	Next text file name.
...			

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

If successful it returns the unique identifier (ID) of the created document in **DataOut.**

Remarks

You may use the object with IPDFXCargs interface to create the text files list as input data. For example (in pseudocode):

```
...
// create empty IPDFXCargs object
IPDFXCargs args;
DoVerb(NULL, ".ARGS", dataIn, dataOut, 0);
args = (IPDFXCargs)dataOut;
dataOut = NULL;
...
// create input files list
args.Add("C:\TextASCII.txt");
args.Add("C:\TextUnicode.log");
args.Add("C:\TextUTF8.ini");
...
// create new document from the plain text files
DoVerb(NULL, "NewDocumentFromText", dataIn, dataOut, 0);
newDocID = dataOut;
...
```

See Also

IPDFXCview::DoVerb, Objects::Operations.NewDocument.FromText

2.2.1.33 OpenDocument

Opens the specified document.

Name

```
OpenDocument
```

Arguments

#	Req.	Type	Description
1	No	BSTR/IStream	Source file name, or valid URL, or Stream Object. The source-URL can contains some special open-actions(see PDF Open Parameters). Some examples: http://www.mysite.com/test.pdf#page=2&zoom=150 file:///C:/test.pdf C:\test.pdf
2	No	BSTR	Open password string.
3	No	BSTR	Name of document for display in control's UI.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

If successful it returns the unique identifier (ID) of the opened document in **DataOut.**

Remarks

If **Flags** contains the PXCVA_Flags::PXCVA_NoUI flag then the argument **#1** should be a valid source name (or URL, or Stream Object).

For example (in pseudocode):

```
...
// show open dialog:
DoVerb(NULL, "OpenDocument", DataIn, DataOut, 0);
// or open document directly:
DoVerb(NULL, "OpenDocument",
       DataIn("C:\Test.pdf"), DataOut, 0);
// or, if document is password protected:
DoVerb(NULL, "OpenDocument",
       DataIn(SafeArray("C:\Test.pdf", "password")), DataOut, 0);
// also open from URL:
DoVerb(NULL, "OpenDocument",
       DataIn("http://www.adobe.com/test.pdf"), DataOut, 0);
// also open from Stream:
DoVerb(NULL, "OpenDocument",
       DataIn(StreamObj), DataOut, 0);
...
```

Note: the Stream Object can be asynchronous (any remote source).
By default all document's content from stream will be placed into temporary file before opening, but you can change this behaviour, see Objects::Documents::UseStreamsDirectly.

See Also

IPDFXCview::DoVerb, IPDFXCview::OpenDocument

2.2.1.34 PrintDocument

Prints a document using parameters previously defined by Objects::Print.

Name

```
PrintDocument
```

Arguments

#	Req.	Type	Description
1	Yes	LONG /BSTR	The unique identifier of the document or full file name of the document.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb, IPDFXCview::PrintDocument

2.2.1.35 RemoveRecentItem

This operation can be used to remove existing item from list of recent documents.

Name

```
RemoveRecentItem
```

Arguments

#	Req.	Type	Description
1	Yes	LONG /BSTR	The unique identifier of opened document or full file name.

Return Value

Returns **S_OK** or an error value otherwise. For obtaining text description of received error code you can use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb

2.2.1.36 RotateDocumentPages

Rotates pages in document using parameters previously defined by Objects::Operations.RotatePages.

Name

```
RotateDocumentPages
```

Arguments

#	Req.	Type	Description
1	Yes	LONG /BSTR	The unique identifier of the document or full file name of the document.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb,
Objects::Operations.RotatePages,
Objects::Documents::<Item>::RotatePages

2.2.1.37 RunJavaScript

Executes the supplied Java Script.

Name

```
RunJavaScript
```

Arguments

#	Req.	Type	Description
1	Yes	BSTR	Specifies the text of Java Script. For more information, see Java Script Support.
2	No	LONG	Specifies the unique identifier of the opened document. This argument can be 0 for the active document. The specified document will be used as the target for the script, referred by ***this*** in the script.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.
If successful then puts the result text of the executed Java Script into the **DataOut.**

See Also

IPDFXCview::DoVerb, IPDFXCview::RunJavaScript

2.2.1.38 SaveDocument

Saves a document in a number of different ways depending on the **Flags** parameter.

Name

```
SaveDocument
```

Arguments

#	Req.	Type	Description
1	Yes	LONG /BSTR	The unique identifier (ID) or full file name of the document.
2	No	BSTR/ IStrea m	The destination file name, or Stream Object.
3	No	LONG	The save flags. For more information, see PXCVA_DocumentSaveFlags.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

Use this operation to save or copy the specified document. To display the standard Save File dialog, pass NULL as the Destination file name. If you want to skip possible error dialogs then you must set the PXCVA_Flags::PXCVA_NoUI flag in the **Flags** argument.

For example (in pseudocode):

```
...
// show save dialog for document "C:\Test.pdf":
DoVerb(NULL, "SaveDocument",
       DataIn("C:\Test.pdf"), DataOut, 0);
// show save dialog for document with ID == 4095:
DoVerb(NULL, "SaveDocument",
       DataIn(4095), DataOut, 0);
// save document directly to "C:\TestCopy.pdf" file
// without UI dialogs:
DoVerb(NULL, "SaveDocument",
       DataIn(SafeArray(4095, "C:\TestCopy.pdf")), DataOut, PXCVA_NoUI);
// save to Stream:
DoVerb(NULL, "SaveDocument",
       DataIn(SafeArray(4095, StreamObj)), DataOut, PXCVA_NoUI);
...
```

The Stream Object can be asynchronous.

See Also

IPDFXCview::DoVerb, IPDFXCview::SaveDocument, Objects::Notifications::BeforeSaveDoc, Objects::Notifications::DocSaved, Objects::Documents::UseStreamsDirectly

2.2.1.39 SaveSettings

Loads all control settings from an external storage.

Name

```
SaveSettings
```

Arguments

#	Req.	Type	Description
1	Yes	BSTR/ IStream	Specifies a valid destination storage: destination file name or pointer to a **IStream** interface.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb, IPDFXCview::SaveSettings

2.2.1.40 ShowStampsCollection

Shows/hides the specified stamps collection.

Name

```
ShowStampsCollection
```

Arguments

#	Req.	Type	Description
1	Yes	BSTR	Name of stamps collection
2	No	LONG	Specify non-zero value to show or zero to hide.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb, Objects::Commenting.HideStdStampsCollections

2.2.1.41 SummarizeDocumentAnnots

Summarizes all comments from a document to a new document(file) using parameters previously defined by Objects::Operations.SummarizeAnnots.

Name

```
SummarizeDocumentAnnots
```

Arguments

#	Req.	Type	Description
1	Yes	LONG /BSTR	The unique identifier of the document or full file name of the document.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb, Objects::Operations.SummarizeAnnots

2.2.2 Objects

Many methods of the interface require a *special name* as input string-argument (**ObjectName** or **Name** argument, see IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty for example).

This argument represents the unique abstract *named object* (simple *property*, *group* of objects, *array* of items).

The common syntax of these strings may be represented as the following:

Syntax

```
<ObjectName> := <Section1>{.<Section2>|.<Section3>|…<SectionN>}
<Section>    := <Name>{[<Index>]}
<Index>      := <{#}Number>|<"<String>">
```

In the above notation:
"{" and **"}"** enclose optional portions;
"|" is the logical OR;
<Name> is a string value;
<Number> is an integer number value, represents a unique identifier, without symbol **"#"** represents a simple order index (zero based) of item in array which is named as **<Name>**
<String> is an string value, represents a unique identifier of item in array which is named as **<Name>**
Some real names by notation for example:

```
...
"General.ApplicationTitle",
"Print.ScaleSimple.AutoRotate",
"ColorManagement["Custom"].RGB",
"View.Bars["File"].Visible",
"Documents[0].Author",
"Documents["C:\Test.pdf"].Title",
"Documents[#4095].View.Panes["Thumbnails"].Visible",
...
```

Named values are supported by get/set operations for many numerical properties (*"<Numeric>/String"* indication in *Type* column of tables). For these properties you can set/get predefined named values (string) instead of numeric values.

For example (in pseudocode) the property with name *"General.AllowAllAccelerators"* supports named values, specified in Booleans:

1.1. get value:

```
...
DoVerb("General.AllowAllAccelerators", "get", DataIn, DataOut, 0);
// or:
GetProperty("General.AllowAllAccelerators", DataOut, 0);
// result:
DataOut == 1; // i.e: all key accelerators are allowed
...
```

1.2. get *named* value:

```
...
DoVerb("General.AllowAllAccelerators", "getNamed", DataIn, DataOut, 0);
// or:
GetProperty("General.AllowAllAccelerators", DataOut, PXCVA_GetNamed);
// result:
DataOut == "True"; // i.e: all key accelerators are allowed
...
```

2.1. set value:

```
...
DoVerb("General.AllowAllAccelerators", "set", DataIn(1), DataOut, 0);
// or:
SetProperty("General.AllowAllAccelerators", DataIn(1), 0);
...
```

2.2. set *named* value:

```
...
DoVerb("General.AllowAllAccelerators", "set", DataIn("True"), DataOut, 0);
// or:
SetProperty("General.AllowAllAccelerators", DataIn("True"), 0);
...
```

Attributes of named objects:

R - readable;
W - writable;
S - storable, i.e. can be saved/restored from/to an external storage (file, memory, stream, etc.).

Top-Level Objects

Top Object Name	Type	Description
ColorManagement	Array [RWS]	Controls the colors of the displayed documents.
Commands	Array [RWS]	Represents an array of all supported named UI-commands.
Commenting	Group [RWS]	Controls for commenting.
Documents	Array [RW]	Represents an array of all opened documents and provides more operations with them.
Export	Group [RWS]	Represents all supported properties for export of documents to an external formats (images, text). See also IPDFXCview::ExportDocument.
Find	Group [RWS]	Contains all supported properties for simple text searching in active document.
Forms	Group [RWS]	Allows to specify settings for PDF forms.
General	Group [RWS]	Accesses the main properties of the ActiveX control.

International	Group [RWS]	Controls UI localization.
Notifications	Group [RWS]	Allows to receive and modify Viewer events.
Operations	Group [RWS]	Allows to specify settings for operations.
PageDisplay	Group [RWS]	Controls the pages of the documents displaying.
Performance	Group [RWS]	Controls the performance of the control.
Print	Group [RWS]	Represents all supported properties for print of documents. See also IPDFXCview::PrintDocument.
Prompts	Group [RWS]	Controls and overrides UI confirms, prompts, messages. See also _IPDFXCviewEvents::OnEvent.
Search	Group [RWS]	Contains all supported properties for extended text searching using search panel in Viewer.
Tools	Group [RWS]	Contains all supported tools and operations with them.
View	Group [RWS]	Controls more attributes of main view: visibility, positions, layouts for main bars, panes, windows. Also provides customizing of main menu and toolbars.

Remarks

All object names are *case-independent*, i.e. the two names *"General.AllowAllAccelerators"* and *"general.allowAllAccelerators"* are equal for all name-input methods.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, IPDFXCview::DoDocumentVerb

2.2.2.1 ColorManagement

Controls the colors of the displayed documents.

Item Template

<Item> - defines item template for each supported color settings profile representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all supported color settings profiles.
Current	String [RWS]	Contains the unique name of the current color settings profile. Can contain two standard values: **"Off"** or **"Custom"**. **"Off"** - the default color settings which will be used from predefined read only item, named as**"ColorManagement["Off"]"**. **"Custom"** - the custom color settings which will be used from the predefined editable item, named as **"ColorManagement["Custom"]"**.

		Default value is **"Off".**
Engine	Integer/ String [RWS]	Specifies the color management engine. For possible named values, see the Color Management Engines. Default value is **0** (Little CMS).
UseBPC	Integer/ String [RWS]	Specifies whether to use black point compensation. For possible named values, see the Booleans. Default value is **0**.

Remarks

This object is the array of items, each item is represented by <Item>. For more information about object names notation, see Object Name Notation.

Example for usage (in pseudocode):

1. get custom RGB working space:

```
...
DoVerb("ColorManagement["Custom"].RGB", "get",
       DataIn, DataOut, 0);
// or:
GetProperty("ColorManagement["Custom"].RGB", DataOut, 0);
...
```

2. set custom RGB working space:

```
...
DoVerb("ColorManagement["Custom"].RGB", "set",
       DataIn("sRGB IEC61966-2.1"), DataOut, 0);
// or:
SetProperty("ColorManagement["Custom"].RGB",
       DataIn("sRGB IEC61966-2.1"), 0);
...
```

3. also enable custom color settings:

```
SetProperty("ColorManagement.Current", DataIn("Custom"), 0);
...
```

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.1.1 <Item>

Defines the item template for each supported color settings profile representation.

Contained Objects

Name	Type	Description
Name	String [RWS]	Defines the special unique name of color settings profile.
RGB	String [RWS]	Defines the name of the **RGB** (Red, Green, Blue) working space.
CMYK	String [RWS]	Defines the name of the **CMYK** (Cyan, Magenta, Yellow, and blacK) working space.
Grayscale	String [RWS]	Defines the name of the **Grayscale** working space.
Flags	Integer [RWS]	For now there is only one possible value equal to **1** which means using output indent. Default value is **0**.

Remarks

The item named as **"Off"** (value of **Name** object above**,** represents default color settings profile) is read only.

See Also

Objects::ColorManagement

2.2.2.2 Commands

Represents an array of all supported named UI-commands and provides more operations with them.

Item Template

<Item> - defines item template for each named UI-command representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains the number of all supported named UI-commands.

Remarks

This object is an array of items, each item is represented by <Item>. For more information about object names notation, see Object Name Notation.

Example for usage (in pseudocode), for enumerate and turn off of all commands:

```
...
// get commands count:
GetProperty("Commands.Count", DataOut, 0);
Count = DataOut;
for i = 0 to Count-1
{
   GetProperty("Commands[" + i + "].ID", DataOut, 0);
   // ...
   GetProperty("Commands[" + i + "].Name", DataOut, 0);
   // ...
   // turn off of command:
   SetProperty("Commands[" + i + "].State", DataIn("Offline"), 0);
}
...
```

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Operations::ExecuteCommand,
Command List

2.2.2.2.1 Command List

This is a list of the UI-items (UI-commands, or other toolbar- menu- items), current as of 14 January 2010.

- **Name** is the unique name of the command.
- **ID** is the corresponding unique ID.
- **Adobe Name** is the corresponding Adobe-name of the command.

-	32944	-	Export
-	33269	-	Form Data
-	36417	-	Exit
-	33280	-	Edit
-	36416	-	About
-	33291	-	Toolbars
-	33290	-	View
-	36415	-	Show
-	36536	-	Delete *Delete Selected Attachments*
-	33292	-	Other Panes
-	36519	-	Show Tabs Thumbnails
-	33293	-	Page Layout
-	33075	-	Rotate View
-	32884	-	Go To
-	33300	-	Document
-	36349	-	Comments
-	36434	-	Show Comments
-	36498	-	Override Document Colors *Toggle Override Document Colors option*
-	33313	-	Zoom Tools
-	33317	-	Link Tools
-	33316	-	Measuring Tools
-	36252	-	Whole Words Only *Find Whole Words Only*
-	32907	-	Display Pages Count of this Document
-	36447	-	Character to/from Gliph Index *Translate Selected Character to/from Gliph Index*

-	36446	-	Character to/from Hex Code *Translate Selected Character to/from Hex Code*
-	33411	-	Text Formatting
-	33314	-	Comment And Markup Tools
-	33311	-	Basic Tools
-	33310	-	Tools
-	36483	-	Line Weights *Show all lines with a one-pixel stroke width regardless of the zoom level*
-	36537	-	Edit Description...
-	36535	-	Add New... *Add New Attachment...*
-	36534	-	Save... *Save Selected Attachment into an external File...*
-	36533	-	Open *Open Selected Attachment*
-	57600	-	New Document...
-	33350	-	Window
-	57670	-	Help
-	36003	-	Cause Exception...
-	36254	-	Include Comments *Searches the Text of Any Comments added to the PDF*
-	33197	-	Look in Sub-Folders
-	36554	-	Include Document Info *Searches the Text in Document Info (in title, subject, author, keywords, etc.)*
-	36552	-	Include Pages Content *Searches the Text of Any Pages*
-	36255	-	Include Bookmarks *Searches the Text of Any Bookmarks*
-	36253	-	Case-Sensitive *Case-Sensitive Search*
-	33373	-	No Properties *No One Object with Properties is Currently Selected*
-	33109	-	Align Center

-	33110	-	Align Right
-	33240	-	Delete *Delete Comment*
-	33233	-	By Type *Group Comments by Type*
-	33234	-	By Author *Group Comments by Author*
-	33238	-	Next *Next Comment*
-	36337	-	Show Dates *Show Modification Dates of Comments*
-	36459	-	Set Current Appearance as Default *Save this Appearance (Colors, Border Style, Opacity, etc.) to Current Comment Style. Saved Appearance will be used for each new comment as Default for Current Comment Style.*
-	36461	-	Locked
-	36460	-	Apply Default Appearance *Apply Saved Appearance (Colors, Border Style, Opacity, etc.) from Current Comment Style*
-	33133	-	Preview
-	36339	-	Show Colors *Show Color of Comments*
-	36338	-	Show Text *Show Text of Comments*
-	36336	-	Show Authors *Show Authors of Comments*
-	36335	-	Show Subjects *Show Subjects of Comments*
-	33256	-	Close This Tab
-	57719	-	Scroll Right
-	33410	-	Comment
-	32936	-	Show All Tabs
-	36539	-	Show All Opened Documents' Thumbnails *View All Opened Documents as Thumbnails*
-	32938	-	Show All Tabs Thumbnails
-	33412	-	Make Snapshot
-	3303	-	Picas

	5		
-	36549	-	Interactive Highlighted Area *Enable/disable the Interactivity of the Highlighted Area*
-	36306	-	Expand All Bookmarks
-	36008	-	New Bookmark *Create a New Bookmark*
-	36026	-	Apply Default Appearance *Apply Default Appearance for Selected Bookmarks*
-	36551	-	Set Initial Zoom Factor... *Set Initial Zoom factor for new "Go to a page in this document" actions...*
-	36012	-	Add *Add a New Action...*
-	36497	-	Add "Open a file"...
-	36018	-	Test All *Test All Actions...*
-	32948	-	Document Title
-	32952	-	File Name
-	32956	-	Day
-	32960	-	Second
-	36276	-	Note Icon
-	36293	-	Create from Clipboard *Create Stamp from Clipboard Image*
-	36298	-	Rename Stamp... *Rename Selected Stamp...*
-	36299	-	Rename Collection... *Rename Selected Collection...*
-	36515	-	Signing
-	33814	-	Edit Description... *Edit Description*
-	36309	-	Reset this Toolbar *Reset this Toolbar to Defaults*
-	36324	-	Choose Image
-	36329	-	Show Image
-	36332	-	Insert Separator After
-	36341	-	Clear Find List

-	36250	-	Highlight Style
-	36357	-	Ratio Unit (From)
-	36356	-	Ratio Value (To)
-	33424	-	Align Bottom Edges
-	33428	-	Distribute Vertically
-	33431	-	Make Same Size
-	36367	-	Redo
-	36371	-	Delete
-	36473	-	Zoom Out
-	36478	-	Rotate 90° CW
-	36542	-	Blend Mode
-	36481	-	Copy Link Location
-	36487	-	Delete Comment Styles *Delete Selected Comment Styles*
-	36488	-	Reset Comment Styles *Reset Selected Comment Styles*
-	36257	-	Edit Subject... *Edit Initial Subject for Selected Comment Style*
-	36486	-	Clone Comment Style *Clone Selected Comment Style*
-	36485	-	Reset Comment *Reset Selected Comment. All custom styles for selected comment type will be removed and only default style will be leaved*
-	36502	-	Open URL in New Browser's Window
-	36501	-	Open URL in New Browser's Tab
-	36482	-	Open URL in Browser
-	36476	-	Fit
-	36479	-	Rotate 180°
-	36477	-	Rotate 90° CCW

-	36475	-	Flip Horizontal
-	36474	-	Flip Vertical
-	36472	-	Zoom In
-	36418	-	Show Comment Styles...
-	36372	-	Select All
-	36370	-	Paste
-	36369	-	Copy
-	36368	-	Cut
-	36366	-	Undo
-	33433	-	Apply Default Appearance *Apply Default Appearance for Selected Links*
-	33432	-	Set Current Link Appearance as Default
-	33430	-	Make Same Height
-	33429	-	Make Same Width
-	33427	-	Distribute Horisontally
-	33422	-	Align Right Edges *Align Rights Edges*
-	33425	-	Align Horizontal Centers
-	33421	-	Align Left Edges
-	33426	-	Align Vertical Centers
-	33423	-	Align Top Edges
-	36355	-	Ratio Value (From)
-	36359	-	Ratio Unit (To)
-	36353	-	Show Cover Page in Facing Mode
-	36352	-	Show Gaps Between Pages
-	36314	-	Clear Shortcut
-	3631	-	Reset Shortcut

	3		
-	36333	-	Properties...
-	36331	-	Insert Separator Before
-	36334	-	Show Preview
-	36330	-	Show Text
-	36328	-	Default Style
-	36327	-	Delete
-	36326	-	Reset
-	36316	-	Reset this Popup-Menu *Reset this Popup-Menu to Defaults*
-	36315	-	Reset Menu Bar *Reset Menu Bar to Defaults*
-	36308	-	Text Properties... *Show Text Properties...*
-	36301	-	Diameter *Eraser Diameter*
-	33813	-	Save Embedded File to Disk... *Save Embedded File to Disk*
-	33812	-	Open Embedded File... *Open Embedded File*
-	36296	-	Delete Collection *Delete Selected Collection*
-	36297	-	Create New Collection... *Create New Stamps Collection...*
-	36304	-	Create from Image... *Create Stamp from Image File...*
-	36303	-	Create from PDF... *Create Stamp from PDF Document...*
-	36294	-	Delete Stamp *Delete Selected Stamp*
-	36292	-	Zoom Out *Reduce Thumbnails*
-	36291	-	Zoom In *Enlarge Thumbnails*
-	33261	-	Text Color
-	36289	-	Line Starting Style
-	36287	-	Line Ending Style
-	33379	-	Opacity *Opacity Level*

-	33231	-	Font Size
-	36281	-	Border Width
-	36457	-	Underline Style
-	36279	-	Border Style
-	36277	-	Zoom
-	36274	-	Stroke Color
-	36273	-	Fill Color
-	36048	-	Set Destination
-	36264	-	Zoom
-	36049	-	Font
-	32961	-	Auto Number
-	32959	-	Minute
-	32958	-	Hour
-	32957	-	Time (HH-MM-SS)
-	32955	-	Month
-	32954	-	Year
-	32953	-	Date (MM-DD-YYYY)
-	32951	-	PAGES
-	32950	-	PAGE
-	32949	-	Page Number
-	32947	-	Document Number
-	32946	-	Document Name
-	36019	-	Reset *Reset List of Actions and Lost Changes*
-	36017	-	Test Selected *Test Selected Actions...*
-	3601	-	Delete

	3		*Delete Selected Actions*
-	36016	-	Edit... *Edit Properties of Selected Action...*
-	36363	-	Add "Open a web link"...
-	36496	-	Add "Go to a page in another document"...
-	36362	-	Add "Go to a page in this document"...
-	36015	-	Down *Move Selected Items Down in the Runtime Order*
-	36014	-	Up *Move Selected Items Up in the Runtime Order*
-	36011	-	Refresh Bookmarks Tree
-	36023	-	Rename *Rename the Selected Bookmark*
-	36256	-	Show Bookmark Icons *Show/Hide Bookmark Icons*
-	36025	-	Wrap Long Bookmark Titles
-	36024	-	Set Current Appearance as Default
-	36010	-	Go to Bookmark
-	36009	-	Ensure Visibility of Corresponding Bookmark
-	36006	-	Delete Bookmarks *Delete Selected Bookmarks*
-	36007	-	Bookmarks Properties... *Show Properties of Selected Bookmarks*
-	36307	-	Collapse All Bookmarks
-	36005	-	Zoom Out Bookmarks *Reduce Bookmarks*
-	36004	-	Zoom In Bookmarks *Enlarge Bookmarks*
-	33414	-	Zoom Out Thumbnails *Reduce Thumbnails*
-	36548	-	Highlight Visible Area *Highlight Visible Area of the Current Page*
-	33413	-	Zoom In Thumbnails *Enlarge Thumbnails*
-	33036	-	Points
-	33034	-	Millimeters

-	33033	-	Inches
-	33032	-	Centimeters
-	36263	-	8,33%
-	36262	-	12,5%
-	57718	-	Scroll Left
-	32937	-	Close Active Tab
-	33255	-	Close All Tabs But This
-	33239	-	Previous *Previous Comment*
-	33241	-	Collapse All *Collapse All Comments*
-	33242	-	Expand All *Expand All Comments*
-	33236	-	By Color *Group Comments by Color*
-	33237	-	By Creation Date *Group Comments by Creation Date*
-	33235	-	By Modification Date *Group Comments by Modification Date*
-	33232	-	By Page *Group Comments by Page*
-	33243	-	Group By
-	33245	-	Add Reply *Add Reply to Comment*
-	33244	-	Properties... *Comment Properties...*
-	33108	-	Align Left
-	36489	-	Rename Comment Style *Rename Selected Comment Style*
-	36511	-	Search By *Search text by provider*
-	36500	-	Manage Recents List...
-	36547	-	Order
-	36513	-	Manage Search Providers...
-	36430	-	Test 1
-	36432	-	Test 3

-	36429	-	Development
-	36431	-	Test 2
-	36514	-	Online Search *Search specified text using internet search provider*
-	33003	-	Pages Properties... *Show Properties of Selected Pages*
-	33267	-	File
About	57664	About	About PDF-XChange Viewer... *Display Program Information, Version Number and Copyright*
ActualSize	32900	ActualSize	Actual Size *Show Actual Page Size (100%)*
AddBookmark	33416	-	Add Bookmark
AddFileAttachment	33815	AddFileAttachment	Attach a File... *Attach a File*
AddLink	33420	-	Add Link
AddNewPageGuides	36523	-	Add New Page Guide(s)...
AddNote	33135	-	Add Note
AddNoteToSelText	33207	-	Add Note to Text
AddReply	33174	-	Add Reply
ApplyDefaultProperties	33134	-	Apply Default Appearance *Apply Saved Appearance (Colors, Border Style, Opacity, etc.) from Current Comment Style*
ApplyDefaultTextFormatting	33260	-	Apply Default Text Formatting *Apply Saved Text Formatting (Font, Font Size, Color, Align, etc.) from Current Comment Style*
AreaTool	36348	MeasurePolygonMenuItem	Area Tool *Area tool to measure area of an object*
ArrangeWindowIcons	57649	-	Arrange Icons *Arrange Icons at the Bottom of the Window*
ArrowTool	33123	Annots:Tool: LineArrowMenuItem	Arrow Tool
CalloutTool	33201	Annots:Tool: FreeTextCalloutMenuItem	Callout Tool
CascadeWindows	57650	Cascade	Cascade *Arrange Windows so they Overlap*
CheckForUpdates	33179	Updates	Check for Updates...

Clear	57632	Clear	Clear *Erase the Selection*
ClearTextFormatting	33168	-	Clear Text Formatting
Close	57602	Close	Close *Close the Active Document*
CloseAll	33038	CloseAll	Close All *Close All Opened Documents*
CloseAllPopups	36439	Annots:ClosePopups	Close All Pop-Ups
CloseAllWindows	32942	CloseAll	Close All Documents
CloseWindow	57682	-	-
CloudTool	33130	Annots:Tool: PolygonCloudMenuItem	Cloud Tool
Continuous	32896	OneColumn	Continuous
ContinuousFacing	32898	TwoColumns	Continuous - Facing
Copy	57634	Copy	Copy *Copy the Selection and Place on the Clipboard*
CopyAsFormattedText	33183	-	Copy As Formatted Text
CopyAsPlainText	33182	-	Copy As Plain Text
CopyAsRichContent	33184	-	Copy As Rich Content
CopyFullFileName	33257	-	Copy Full File Name *Copy Full File Name of the Active Document to Clipboard*
CropPages	33082	CropPages	Crop Pages...
CrossOutSelText	33209	-	Cross Out Text
CrossOutTextTool	33223	Annots:Tool: StrikeOutMenuItem	Cross Out Text Tool *Cross Out the Selected Text*
Cut	57635	Cut	Cut *Cut the Selection and Place on the Clipboard*
Delete	33136	Delete	Delete *Delete the Selection*
DeleteAllPageGuides	36525	-	Delete All Page Guides
DeleteGuidesOnCurrentPage	36524	-	Delete Guides on Current Page
DeletePages	33081	DeletePages	Delete Pages...
DeleteReply	33176	-	Delete Reply

DeleteSelectedPages	33819	-	Delete Selected Pages
DeselectAll	33248	DeselectAll	Deselect All
DistanceTool	36346	MeasureLineMenuItem	Distance Tool *Distance tool to measure distance between two points*
DocumentOptions	33370	-	Document Options *Contains Viewing, Editing, Manipulating, etc. Options for this Document*
DocumentProperties	33270	GeneralInfo	Document Properties... *Show Properties of the Active Document*
EraserTool	36300	Annots:Tool: EraseInkMenuItem	Eraser Tool *Eraser Tool erases only those parts that were drawn by a Pencil Tool*
ExportAllSettings	36409	-	Export All Settings to Data File... *Export All Application's Settings to Data File...*
ExportAnnots	36351	-	Export Comments to Data File...
ExportFormData	33073	AcroForm: FormData_ExportData	Export Data from Form...
ExportToImage	32945	ImageConversion:Export	Export to Image... *Export Pages to Image(s)...*
ExtractPages	36038	ExtractPages	Extract Pages...
Facing	32897	TwoPages	Facing
FileAttachmentTool	33224	Annots:Tool: FileAttachmentMenuItem	File Attachment Tool *Attach a File as Comment. Click the page to pin a file attachment to that place on the page.*
Find	57636	Find	Find *Find the Specified Text*
FindCaseSensitive	32995	-	Case-Sensitive *Case-Sensitive Search*
FindEdit	32990	FindEdit	Find the Specified Text
FindIncludeBookmarks	32997	-	Include Bookmarks *Searches the Text of Any Bookmarks*
FindIncludeComments	32996	-	Include Comments *Searches the Text of Any Comments added to the PDF*
FindIncludePagesContent	36553	-	Include Pages Content *Searches the Text of Any Pages*
FindNext	32992	FindAgain	Find Next *Finds the Next Occurrence*
FindOptions	32993	-	Find Options

FindPrevious	32991	FindPrevious	Find Previous *Finds the Previous Occurrence*
FindWholeWords	32994	-	Whole Words Only *Find Whole Words Only*
FirstPage	32885	FirstPage	First Page *Go to First Page*
FitPage	32901	FitPage	Fit Page *Fit Page by Current View*
FitVisible	32903	FitVisible	Fit Visible *Fit Page by Visible Content*
FitWidth	32902	FitWidth	Fit Width *Fit Page Width by Current View*
FlattenAnnots	36340	-	Flatten Comments...
FlipLine	33115	-	Flip Line
FontsList	33230	-	-
FullScreenMode	33014	FullScreen	Full Screen *Maximizes Window to Full Screen*
FullScreenModeExit	33020	-	Full Screen Exit *Restores Window from Full Screen*
FullSearch	32987	FindSearch	Search *Show/Hide Full Search Pane*
GoBack	32316	GoBack	Back
GoBack	32890	GoBack	Previous View *Go To Previous View*
GoBackDoc	32892	GoBackDoc	Previous Document
GoForward	32317	GoForward	Forward
GoForward	32891	GoForward	Next View *Go To Next View*
GoForwardDoc	32893	GoForwardDoc	Next Document
GoToHomePage	33141	-	Home Page
GoToNextSearchOccur	32979	-	Go to Next Search Occurrence
GoToPage	32889	GoToPage	Page...
GoToPrevSearchOccur	32978	-	Go to Previous Search Occurrence
HandTool	32613	HandMenuItem	Hand Tool
HelpContents	33145	HelpUserGuide	Contents
HideAllBars	36557	-	Hide All Bars *Hide all bars from user's set*

HideAllComments	36435	Annots:Hide All Comments	Hide All Comments
HighlightSelText	33206	-	Highlight Text
HighlightTextTool	33221	Annots:Tool: HighlightMenuItem	Highlight Text Tool *Highlight the Selected Text*
HilightAllCheckBoxes	36465	-	Check Boxes *Click Here to Highlight All Check Boxes*
HilightAllComboBoxes	36464	-	Combo Boxes *Click Here to Highlight All Combo Boxes*
HilightAllFormFields	36463	-	All Fields *Click Here to Highlight All Form Fields*
HilightAllListBoxes	36466	-	List Boxes *Click Here to Highlight All List Boxes*
HilightAllPushButtons	36467	-	Push Buttons *Click Here to Highlight All Push Buttons*
HilightAllRadioButtons	36468	-	Radio Buttons *Click Here to Highlight All Radio Buttons*
HilightAllSignatures	36470	-	Signature Fields *Click Here to Highlight All Signature Fields*
HilightAllTextBoxes	36469	-	Text Boxes *Click Here to Highlight All Text Boxes*
HilightFormFields	33018	-	Highlight Form Fields *This Document Contains Form; Click Here to Highlight Form Fields by Filter*
HilightRequiedFieldsOnly	36471	-	Required Fields Only *Click Here to Highlight Required Fields Only*
ImportAllSettings	36408	-	Import All Settings from Data File... *Import All Application's Settings from Data File...*
ImportComments	36503	-	Import Comments...
ImportFormData	33074	AcroForm: Forms_ImportData	Import Data to Form...
InsertEmptyPages	36305	-	Insert Empty Pages...
InsertPages	36037	InsertPages	Insert Pages...
JSConsole:Clear	36028	-	Clear *Clear Console Output*
JSConsole:GoTo	36035	-	Go to line...
JSConsole: RunScript	36027	-	Run *Run Script*
JSConsole:	3603	-	Java Script Options

ShowOptions	6		*Show Java Script Options*
KeepToolSelected	33229	-	Keep Tool Selected
LastPage	32888	LastPage	Last Page *Go to Last Page*
LineTool	33124	Annots:Tool: LineMenuItem	Line Tool
LockAllCmdBars	36373	-	Lock/Unlock All Command Bars
LoupeTool	32609	LoupeMenuItem	Loupe Tool
MakeTextFormattingAsDefault	33259	-	Set Current Text Formatting as Default *Save this Text Formatting (Font, Font Size, Color, Align, etc.) to Current Comment Style. Saved Text Formatting will be used for each new comment as Default for Current Comment Style.*
MeasureDeleteLabel	33801	-	Delete Label
MeasureEditLabel	33802	-	Edit Label
MeasureExportToCSV	33811	-	Export Measurements To CSV File *Export Measurements*
MeasureInfoWindow	33807	-	Show Measure Info Window
MeasureRestoreCaptionPos	33800	-	Restore Caption Placement
MeasureSetLabel	33806	-	Change Markup Label
MeasureSetScale	33805	-	Change Scale Ratio
NewBlankDocument	36364	-	From Blank Page...
NewDocumentFromImage	36375	-	From Image File...
NewDocumentFromRTF	36493	-	From Rich Text Format (RTF) File...
NewDocumentFromScanner	36462	-	From Scanner...
NewDocumentFromText	36365	-	From Text File...
NextPage	32887	NextPage	Next Page *Go to Next Page*
NextWindow	57680	-	-
Open	57601	Open	Open... *Open an Existing Document...*
OpenAllPopups	36438	Annots:OpenPopups	Open All Pop-Ups

OpenContainingFolder	33258	-	Open Containing Folder... *Open the Source Folder of the Active Document*
OpenPopup	33122	-	Open Pop-Up Note
OpenURL	33272	-	Open from URL... *Open Document from URL...*
Order:BackOne	36546	-	Back One
Order:ForwardOne	36545	-	Forward One
Order:ToBackground	36544	-	To Background *Move to Background*
Order:ToForeground	36543	-	To Foreground *Move to Foreground*
OvalTool	33126	Annots:Tool:CircleMenuItem	Oval Tool
PageNumber	32909	-	Display Current Page Number in this Document
PanAndZoomTool	33103	PanAndZoomMenuItem	Pan and Zoom Tool
Paste	57637	Paste	Paste *Insert Clipboard Contents*
PencilTool	33129	Annots:Tool:InkMenuItem	Pencil Tool
PerimeterTool	36347	MeasurePolylineMenuItem	Perimeter Tool *Perimeter tool to measure perimeter of an object*
PlaceSignature	36517	DIGSIG:PlaceSigPullRight	Place Signature
PolyCancel	33139	-	Cancel
PolyComplete	33140	-	Complete
PolygonTool	33127	Annots:Tool:PolygonMenuItem	Polygon Tool
PolyLineTool	33128	Annots:Tool:PolyLineMenuItem	Polygon Line Tool
Preferences	32906	GeneralPrefs	Preferences... *Display Application Preferences...*
PrevPage	32886	PrevPage	Previous Page *Go to Previous Page*
PrevWindow	57681	-	-
Print	57607	Print	Print... *Print the Active Document*
ProblemReport	33146	-	Report a Problem...
QuadLinkTool	36344	-	Quadrilateral Link Tool

Quit	57665	Quit	Exit *Quit and Prompt to Save any Changed Documents*
RecentsList	33268	-	Recent Files
RectangleTool	33125	Annots:Tool: SquareMenuItem	Rectangle Tool
RectLinkTool	36343	-	Rectangle Link Tool
Redo	57644	Redo	Redo *Redo the Previously Undone Action*
ReplacePages	36039	ReplacePages	Replace Pages...
ReplaceSelText	33210	-	Replace Text
ResetAllSettings	36374	-	Reset All Settings to Defaults... *Reset All Application's Settings to Defaults...*
ResetPopupLocation	33121	-	Reset Pop-Up Note Location
RotateCCW	33079	RotateCCW	Counterclockwise (CCW) *Rotate View Counterclockwise (CCW)*
RotateCW	33078	RotateCW	Clockwise (CW) *Rotate View Clockwise (CW)*
RotatePages	33080	RotatePages	Rotate Pages...
RotateSelectedPages180	33817	-	Rotate Selected Pages 180°
RotateSelectedPages270	33818	-	Rotate Selected Pages 270°
RotateSelectedPages90	33816	-	Rotate Selected Pages 90°
Save	57603	Save	Save *Save the Active Document*
SaveAs	57604	SaveAs	Save As... *Save the Active Document with a New Name*
SaveCopyAs	57611	-	Save Copy As... *Save the Active Document to a New File*
SelectAll	57642	SelectAll	Select All
SelectTool	32620	SelectMenuItem	Select Tool *Select Text/Objects Tool*
SendMail	57612	AcroSendMail:SendMail	Send by E-mail... *Attach the Active Document to a New E-mail Message*
SetPropertiesAsDefault	33118	-	Set Current Appearance as Default *Save this Appearance (Colors, Border Style, Opacity, etc.) to Current Comment Style. Saved Appearance*

			will be used for each new comment as Default for Current Comment Style.
SetTransitions	36040	SetTransitions	Pages Transitions...
SetupMeasurement	36527	-	Setup Measurement...
ShowAllBars	36556	-	Show All Bars *Show all bars from user's set*
ShowAllComments	36436	Annots:Show All Comments	Show All Comments
ShowAllTypes	36440	Annots:Show all Types	All Types
ShowAttachments	36445	Annots:Attachments	Attachments
ShowBrokenInfo	33369	-	Show Broken Info *This Document is Broken; Click Here to Show Details*
ShowCmdCustomizeDialog	36000	-	Customize... *Display Dialog allowing Bar Customization*
ShowCommentsByType	36437	Annots:Show Types	Show by Type
ShowCommentStylesPalette	36420	-	Show Comments Styles Palette
ShowDrawings	36442	Annots:Drawing	Drawing Markups
ShowGrid	36520	ShowGrid	Show Grid
ShowGuides	36521	ShowGuides	Show Guides
ShowNotes	36441	Annots:Comments	Notes
ShowPDFAInfo	33374	-	Show PDF/A Info *This Document is a PDF/A compliant file; Click Here to Show Details*
ShowRulers	36522	ShowRulers	Show Rulers
ShowStamps	36444	Annots:Stamp	Stamps
ShowStampsPalette	33228	Annots: ShowStampsPalette	Show Stamps Palette
ShowTexts	36443	Annots:Markup	Text Editing Markups
ShowXFAInfo	33375	-	Show XFA Info *This Document contains an XFA Form; Click Here to Show Details*
SignDocument	36516	DIGSIG:NextSigPullRight	Sign Document *Digitally Sign a Document*
SinglePage	32895	SinglePage	Single Page

SnapOnOff	36532	-	Snap *Snap On/Off*
Snapshot.CopyPageSelection	33383	-	Copy Page Selection
Snapshot.SelectEntirePage	33384	-	Select Entire Page
SnapshotTool	32614	SelectGraphicsMenuItem	Snapshot Tool
SnapToGrid	36529	SnapToGrid	Snap to Grid
SnapToGuides	36530	SnapToGuides	Snap To Guides
SnapToObjects	36531	SnapToObjects	Snap To Objects
SplitWindow	57653	SplitWindow	Split *Split the Active Window into Panes*
StampTool	33131	Annots:Tool:StampMenuItem	Stamp Tool
StartFullSearch	32977	-	Start Full Search
StickyNoteTool	33132	Annots:Tool:TextMenuItem	Sticky Note Tool
StopFullSearch	32976	-	Stop Full Search
SummarizeAnnots	36350	-	Summarize Comments...
SuportForum	33147	AccessOnline	Support Forum
TextBoxTool	33202	Annots:Tool:FreeTextMenuItem	Text Box Tool
TextEditor:Bold	33104	-	Bold
TextEditor:Italic	33105	-	Italic
TextEditor:Strikeout	33107	-	Strikeout
TextEditor:Subscript	33171	-	Subscript
TextEditor:Superscript	33170	-	Superscript
TextEditor:Underline	33106	-	Underline
TileWindowsHorizontal	57651	TileHorizontal	Tile Horizontally *Arrange Windows as Non-Overlapping tiles*
TileWindowsVertical	57652	TileVertical	Tile Vertically *Arrange Windows as Non-Overlapping tiles*
TipAttachmentsContains	36538	-	Show Attachments Pane *This Document Contains the File*

			Attachments; Click Here to Show/Hide Attachments Pane
TipFieldsContains	33372	-	Show Fields Pane *This Document Contains the Form; Click Here to Show/Hide Form Fields Pane*
TipLayersContains	33371	-	Show Layers Pane *This Document Contains Layers; Click Here to Show/Hide Layers Pane*
ToggleAllBars	33015	-	Show/Hide All Bars *Show/Hide All Bars from user's set*
ToggleAllToolbars	33016	-	Show/Hide All Toobars *Show/Hide All Toobars from user's set*
ToggleAnnotLockState	36001	-	Locked
ToggleAttachmentsPane	33205	ShowHideFileAttachment	Attachments *Show/Hide Attachments Pane*
ToggleBookmarksPane	32910	ShowHideBookmarks	Bookmarks *Show/Hide Bookmarks Pane*
ToggleCommentingBar	33339	CommentingMainToolBar	Show Comment And Markup Toolbar
ToggleCommentingBar	33138	ShowHideToolbarCommenting	Comment And Markup Toolbar *Show/Hide Comments And Markup Toolbar*
ToggleCommentsPane	33340	ShowHideAnnotManager	Show Comments List *Show/Hide Comments List*
ToggleCommentsPane	33204	ShowHideAnnotManager	Comments *Show/Hide Comments Pane*
ToggleDocumentOptionsBar	33262	-	Document Options Toolbar *Show/Hide Document Options Toolbar*
ToggleFieldsPane	33008	ShowHideFields	Fields *Show/Hide Fields Pane*
ToggleFileBar	32908	ShowHideToolbarFile	File Toolbar *Show/Hide File Toolbar*
ToggleFindBar	33000	ShowHideToolbarFind	Find Toolbar *Show/Hide Find Toolbar*
ToggleJSConsole	33187	-	Java Script Console *Show/Hide Java Script Console*
ToggleLaunchBar	33265	-	Launch Toolbar *Show/Hide Launch Applications Toolbar*
ToggleLayersPane	33006	ShowHideOptCont	Layers *Show/Hide Layers Pane*
ToggleLinksBar	36342	-	Links Editor Toolbar *Show/Hide Links Editor Toolbar*
ToggleMeasuringBar	36345	MeasuringMainToolBar	Measuring Toolbar *Show/Hide Measuring Toolbar*
ToggleMenuBar	33009	-	Menu Bar *Show/Hide Menu Bar*
ToggleNavigationTabsBar	32940	-	Navigation Tabs *Show/Hide Navigation Tabs*

TogglePagesLayoutBar	33264	ShowHideToolbarPageDisplay	Pages Layout Toolbar *Show/Hide Pages Layout Toolbar*
TogglePagesNavigationBar	33263	ShowHideToolbarNavigation	Pages Navigation Toolbar *Show/Hide Pages Navigation Toolbar*
TogglePropertiesBar	33225	PropertyToolbar	Properties Toolbar *Show/Hide Properties Toolbar*
ToggleRotateViewBar	32929	ShowHideToolbarRotateView	Rotate View Toolbar *Show/Hide Rotate View Toolbar*
ToggleStandardBar	32924	-	Standard Toolbar *Show/Hide Standard Toolbar*
ToggleStatusBar	59393	-	Status Bar *Show/Hide Status Bar*
ToggleTextFormattingBar	33382	-	Show/Hide Text Formatting Toolbar
ToggleThumbnailsPane	32912	ShowHideThumbnails	Pages Thumbnails *Show/Hide Pages Thumbnails Pane*
ToggleXFAPane	33010	ShowHideXFA	XFA Structure *Show/Hide XFA Pane*
ToggleZoomBar	32925	-	Zoom Toolbar *Show/Hide Zoom Toolbar*
ToolProperties	33116	-	Properties... *More Properties...*
TypewriterTool	33226	Annots:Tool:FreeTextTypewriterMenuItem	Typewriter Tool
UnderlineSelText	33208	-	Underline Text
UnderlineTextTool	33222	Annots:Tool:UnderlineMenuItem	Underline Text Tool *Underline the Selected Text*
Undo	57643	Undo	Undo *Undo the Last Action*
ValidateAllSignatures	36518	DIGSIG:ValidateAll	Validate All Signatures
Zoom100	33062	-	100%
Zoom1200	33057	-	1200%
Zoom150	33061	-	150%
Zoom1600	36260	-	1600%
Zoom200	33060	-	200%
Zoom2400	33056	-	2400%
Zoom25	33065	-	25%
Zoom3200	33055	-	3200%
Zoom400	3305	-	400%

	9		
Zoom50	33064	-	50%
Zoom6400	33054	-	6400%
Zoom75	33063	-	75%
Zoom800	33058	-	800%
ZoomIn	32904	ZoomViewIn	Zoom In
ZoomInTool	32610	ZoomInMenuItem	Zoom In Tool
ZoomOut	32905	ZoomViewOut	Zoom Out
ZoomOutTool	32611	ZoomOutMenuItem	Zoom Out Tool
ZoomTo	32899	ZoomTo	Zoom To... *Select Magnification Level for Active Document*

2.2.2.2.2 <Item>

Defines item template for each named UI-command representation.

Contained Objects

Name	Type	Description
ID	Integer [R]	Defines special unique identifier of the command.
Name	String [R]	Defines special unique name of the command. Can be empty.
Title	String [R]	Contains current UI-title of the command.
Tooltip	String [R]	Contains current UI-tooltip of the command.
State	String [RWS]	Determines the state of the command. For supported values, see Command States.
AdobeName	String [R]	Defines special unique Adobe-name of the command. Can be empty.
Shortcut	Group [RWS]	Controls keyboard shortcut(accelerator) for the command.
ShortcutText	String [R]	Returns the shortcut string for displaying in UI, such as: "Ctrl+O", "Ctrl+Alt+S", "Ctrl+P".

See Also

Objects::Commands

2.2.2.2.2.1 Shortcut

Controls keyboard shortcut(accelerator) for the UI-command.

Contained Objects

Name	Type	Description
Modifiers	Integer [RWS]	Defines accelerator flags. It can be a combination of the following values from the Shortcut Modifiers. Put **-1**(or **128**) to reset the shortcut to defaults. If returned value contains the **128** flag then shortcut is default. ***Note:*** *in previous versions control returned the* ***-1*** *for all the default shortcuts. But, in this case, you couldn't obtain the real default shortcut(key, key type, modifiers), therefore in the new version it has been changed. Please update your code in this case.*
Key	Integer [RWS]	Defines code depended with **Type** member. For example: 'A', 'S', 'C', 0x21(VK_PRIOR).
Type	Integer/ String [RWS]	Specifies the **Key** type: virtual key code or ASCII code. For possible named values, see the Shortcut Key Types.
Use	Integer/ String [RWS]	Determines shortcut usage for the command. Put **0**(false) for disable the shortcut. For possible named values, see the Booleans.

See Also

Objects::Commands::<Item>

2.2.2.3 Commenting

Controls for commenting.

Contained Objects

Name	Type	Description
Area	Group [RWS]	Specifies area commenting preferences.
Arrow	Group [RWS]	Specifies arrow commenting preferences.
Callout	Group [RWS]	Specifies callout commenting preferences.
Cloud	Group [RWS]	Specifies cloudy commenting preferences.
Distance	Group [RWS]	Specifies distance commenting preferences.
FileAttachment	Group [RWS]	Specifies file attachment commenting preferences.
Highlight	Group [RWS]	Specifies highlight commenting preferences.
Line	Group [RWS]	Specifies line commenting preferences.
Link	Group [RWS]	Specifies link commenting preferences.
Oval	Group [RWS]	Specifies oval commenting preferences.

Pencil	Group [RWS]	Specifies pencil commenting preferences.
Perimeter	Group [RWS]	Specifies perimeter commenting preferences.
Polygon	Group [RWS]	Specifies polygon commenting preferences.
Polyline	Group [RWS]	Specifies polyline commenting preferences.
Rect	Group [RWS]	Specifies rectangle commenting preferences.
Stamp	Group [RWS]	Specifies stamp commenting preferences.
StickyNote	Group [RWS]	Specifies stickynote commenting preferences.
StrikeOut	Group [RWS]	Specifies strikeout commenting preferences.
TextBox	Group [RWS]	Specifies textbox commenting preferences.
Typewriter	Group [RWS]	Specifies typewriter commenting preferences.
Underline	Group [RWS]	Specifies underline commenting preferences.
CopySelTextToDrawingPopup	Integer/ String [RWS]	Specifies to copy encircled text into Drawing comment popups automatically. For possible named values, see the Booleans. Default value is **0** (false).
CopySelTextToHilightPopup	Integer/ String [RWS]	Specifies to copy selected text into Highlight, Cross-Out, and Underline comment popups automatically. For possible named values, see the Booleans. Default value is **0** (false).
ShowTextIndicators	Integer/ String [RWS]	Specifies to show special text indicators for comments. For possible named values, see the Booleans. Default value is **1** (true).
ShowTooltips	Integer/ String [RWS]	Specifies to show tooltips for comments. For possible named values, see the Booleans. Default value is **1** (true).
AnnotCommonSubj	String [RWS]	Common subject text for styles where **DefSubjMode** property is set to **"Common"**.
SimpleFocusApp	Integer/ String [RWS]	Toggles between animated and simple appearance of focus-rectangle for annotations and form-fields. For possible named values, see the Booleans. Default value is **0** (false).
HideStdStampsCollections	Integer/ String [RWS]	Hides/Shows all standard(built-in) stamps collections. For possible named values, see the Booleans. Default value is **0** (false).

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.1 Area

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.1.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for area comment:
GetProperty("Commenting.Area.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Area.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting area style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.

Name	String [RWS]	Defines name of comment style.
FColor	Integer/ String [RWS]	Specifies fill color for the comment. For possible named values, see the Colors. Default value is **-1** (none).
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
Border	Group [RWS]	Specifies border for the comment.
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.2 Arrow

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.2.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for arrow comment:
GetProperty("Commenting.Arrow.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Arrow.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting arrow style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
Start	Integer/ String [RWS]	Specifies line starting style. For possible named values, see the Line Ending . Default value is **0** (none)
End	Integer/ String [RWS]	Specifies line endings style. For possible named values, see the Line Ending . Default value is **4** (Open Arrow)
FColor	Integer/ String [RWS]	Specifies fill color for the comment. For possible named values, see the Colors. Default value is **-1** (none).
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
Border	Group [RWS]	Specifies border for the comment.

DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.3 Callout

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.3.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for callout comment:
GetProperty("Commenting.Callout.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Callout.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting callout style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
End	Integer/ String [RWS]	Specifies line ending style. For possible named values, see the Line Ending .
FColor	Integer/ String [RWS]	Specifies fill color for the comment. For possible named values, see the Colors. Default value is **RGB(255,255,255)** (black).
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
Border	Group [RWS]	Specifies border for the comment.
TextFormat	Group [RWS]	Specifies text format for the comment.
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.4 Cloud

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description

Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.4.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for cloudy comment:
GetProperty("Commenting.Cloud.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Cloud.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting cloudy style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
FColor	Integer/ String [RWS]	Specifies fill color for the comment. For possible named values, see the Colors. Default value is **-1** (none).
SColor	Integer/	Specifies stroke color for the comment.

	String [RWS]	For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
Border	Group [RWS]	Specifies border for the comment.
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.5 Distance

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.5.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for distance comment:
GetProperty("Commenting.Distance.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Distance.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting distance style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
Start	Integer/ String [RWS]	Specifies line starting style. For possible named values, see the Line Ending . Default value is **5** (ClosedArrow).
End	Integer/ String [RWS]	Specifies line endings style. For possible named values, see the Line Ending . Default value is **5** (ClosedArrow).
FColor	Integer/ String [RWS]	Specifies fill color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
Border	Group [RWS]	Specifies border for the comment.
LeaderExtended	Double [RWS]	Specifies the length of the leader line extensions that expend from the line proper 180 degrees from the leader lines. Default value is **5.0**.
LeaderOffset	Double [RWS]	Specifies the length of the leader line offset, which is the amount of empty space between the endpoint's of the annotation and the

		beginning of the leader lines. Default value is **0.0**.
LeaderLength	Double [RWS]	Specifies the length of leader lines in default user space that extend from each endpoint of the line perpendicular to line itself. Default value is **15.0**.
ShowCaption	Integer/ String [RWS]	Specifies visibility of caption. For possible named values, see the Booleans. Default value is **1** (true).
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.6 FileAttachment

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.6.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for attachment comment:
GetProperty("Commenting.FileAttachment.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.FileAttachment.Styles[" + i + "].ID", DataOut, 0
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting file attachment style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
Type	Integer/ String [RWS]	Specifies ID of an icon to be used in displaying the file attachment. For possible named values, see the File Attachment Icon Types. Default value is 3 (Comment).
FColor	Integer/ String [RWS]	Specifies fill color for the comment. For possible named values, see the Colors. Default value is **RGB(64,85,255)**.
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.7 Highlight

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description

Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.7.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for highlight comment:
GetProperty("Commenting.Highlight.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Highlight.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting highlight style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(255,255,0)** (yellow).
Opacity	Double	Specifies the opacity of the comment.

	[RWS]	Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **1** (Multiply).
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.8 Line

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.8.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for line comment:
GetProperty("Commenting.Line.Styles.Count", DataOut, 0);
Count = DataOut;
...
```

```
for i = 0 to Count-1
{
  GetProperty("Commenting.Line.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting line style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
Start	Integer/ String [RWS]	Specifies line starting style. For possible named values, see the Line Ending . Default value is **0** (none)
End	Integer/ String [RWS]	Specifies line endings style. For possible named values, see the Line Ending . Default value is **0** (none)
FColor	Integer/ String [RWS]	Specifies fill color for the comment. For possible named values, see the Colors. Default value is **-1** (none).
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
Border	Group [RWS]	Specifies border for the comment.
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.9 Link

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.9.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for link comment:
GetProperty("Commenting.Link.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Link.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting link style.

Contained Objects

Name	Type	Description

ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(0,0,0)** (black).
Border	Group [RWS]	Specifies border for the comment.
HighlightMode	Integer/ String [RWS]	Specifies highlight mode for links. For possible named values, see the Link Highlight Mode. Default value is **1** (invert).
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.10 Oval

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.10.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for oval comment:
GetProperty("Commenting.Oval.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Oval.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting oval style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
FColor	Integer/ String [RWS]	Specifies fill color for the comment. For possible named values, see the Colors. Default value is **-1** (none).
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
Border	Group [RWS]	Specifies border for the comment.
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.11 Pencil

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.11.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for pencil comment:
GetProperty("Commenting.Pencil.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Pencil.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting pencil style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.

Name	String [RWS]	Defines name of comment style.
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
Border	Group [RWS]	Specifies border for the comment.
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.12 Perimeter

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.12.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for perimeter comment:
GetProperty("Commenting.Perimeter.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Perimeter.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting perimeter style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
Start	Integer/ String [RWS]	Specifies line starting style. For possible named values, see the Line Ending . Default value is **5** (ClosedArrow).
End	Integer/ String [RWS]	Specifies line endings style. For possible named values, see the Line Ending . Default value is **5** (ClosedArrow).
FColor	Integer/ String [RWS]	Specifies fill color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
Border	Group [RWS]	Specifies border for the comment.
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/	Specifies mode for default subject.

	String [RWS]	For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.13 Polygon

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.13.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for polygon comment:
GetProperty("Commenting.Polygon.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Polygon.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting polygon style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
FColor	Integer/ String [RWS]	Specifies fill color for the comment. For possible named values, see the Colors. Default value is **-1** (none).
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
Border	Group [RWS]	Specifies border for the comment.
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.14 Polyline

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.14.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for polyline comment:
GetProperty("Commenting.Polyline.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Polyline.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting polyline style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
Start	Integer/ String [RWS]	Specifies line starting style. For possible named values, see the Line Ending . Default value is **0** (none)
End	Integer/ String [RWS]	Specifies line endings style. For possible named values, see the Line Ending . Default value is **4** (Open Arrow)
FColor	Integer/ String [RWS]	Specifies fill color for the comment. For possible named values, see the Colors. Default value is **-1** (none).
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
Opacity	Double	Specifies the opacity of the comment.

	[RWS]	Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
Border	Group [RWS]	Specifies border for the comment.
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.15 Rect

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.15.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for rect comment:
GetProperty("Commenting.Rect.Styles.Count", DataOut, 0);
```

```
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Rect.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting rectangle style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
FColor	Integer/ String [RWS]	Specifies fill color for the comment. For possible named values, see the Colors. Default value is **-1** (none).
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
Border	Group [RWS]	Specifies border for the comment.
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.16 Stamp

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.16.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for stickynote comment:
GetProperty("Commenting.StickyNote.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Stamp.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting text box style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
Type	String [RWS]	Specifies type of the stamp comment. For possible values, see remarks.
SColor	Integer/	Specifies stroke color for the comment.

	String [RWS]	For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
Border	Group [RWS]	Specifies border for the comment.
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

Remarks

Next stamp types is possible:
Approved, AsIs, Confidential, Departmental, Draft, Experimental, Expired, Final, ForComment, ForPublicRelease, NotApproved, NotForPublicRelease, Sold, TopSecret.

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.17 StickyNote

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.17.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for stickynote comment:
GetProperty("Commenting.StickyNote.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.StickyNote.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting stickynote style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
Type	Integer/ String [RWS]	Specifies ID of an icon to be used in displaying the sticky note. For possible named values, see the Sticky Note Icon Types. Default value is 3 (Comment).
FColor	Integer/ String [RWS]	Specifies fill color for the comment. For possible named values, see the Colors. Default value is **RGB(255,255,0)** (yellow).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.18 StrikeOut

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.18.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for strikeout comment:
GetProperty("Commenting.StrikeOut.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Strikeout.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting strikeout style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (Red).

Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.19 TextBox

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.19.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for text box comment:
GetProperty("Commenting.TextBox.Styles.Count", DataOut, 0);
Count = DataOut;
```

```
...
for i = 0 to Count-1
{
  GetProperty("Commenting.TextBox.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting text box style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
FColor	Integer/ String [RWS]	Specifies fill color for the comment. For possible named values, see the Colors. Default value is **RGB(255,255,255)** (white).
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
Border	Group [RWS]	Specifies border for the comment.
TextFormat	Group [RWS]	Specifies text format for the comment.
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.20 Typewriter

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.20.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for typewriter comment:
GetProperty("Commenting.Typewriter.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Typewriter.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting typewriter style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
FColor	Integer/ String	Specifies fill color for the comment. For possible named values, see the Colors.

	[RWS]	Default value is **-1** (none).
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(0,0,0)** (black).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
Border	Group [RWS]	Specifies border for the comment.
TextFormat	Group [RWS]	Specifies text format for the comment.
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.3.21 Underline

Accesses the properties of the commenting.

Contained Objects

Name	Type	Description
Styles	Array [RWS]	Defines item style representation.

See Also

Objects::Commenting

2.2.2.3.21.1 Styles

This object represents array of all comments styles and provides additional operations with them.

Item Template

<Item> - defines item template for each style representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all accessible styles.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

```
...
// Enumerate all styles for underline comment:
GetProperty("Commenting.Underline.Styles.Count", DataOut, 0);
Count = DataOut;
...
for i = 0 to Count-1
{
  GetProperty("Commenting.Underline.Styles[" + i + "].ID", DataOut, 0);
   ...
}
...
```

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Defines item template for each commenting underline style.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Specifies the unique ID of the comment style.
Name	String [RWS]	Defines name of comment style.
SColor	Integer/ String [RWS]	Specifies stroke color for the comment. For possible named values, see the Colors. Default value is **RGB(0,255,0)** (green).
Opacity	Double [RWS]	Specifies the opacity of the comment. Default value is **1.0** (100% opacity).
BlendMode	Integer/ String [RWS]	Specifies the blend mode for the comment. For possible named values, see the Blend Modes. Default value is **0** (Normal).
DefSubj	String [RWS]	Specifies default subject text for comment. This value will be used only if "DefSubjMode" is set to "Custom".
DefSubjMode	Integer/ String [RWS]	Specifies mode for default subject. For possible named values, see the Subject Modes. Default value is **0** (Default).

See Also

Objects::Commenting,
IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.4 Documents

This object represents array of all opened documents and provides additional operations with them.

Item Template

<Item> - defines item template for each opened document representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all opened documents.
Active	Integer [RW]	Contains the unique identifier of the active document.
ContentMonitor	Integer/String [RWS]	Controls a special monitor that watches the display completeness of the page content. For possible named values, see the Content Monitor States. Default value is **0** (Off).
UseStreamsDirectly	Integer/String [RWS]	Specifies using passed Stream Object directly for opening without placing content to disk. Also, if it means **1** that passed stream object will be captured by the control to have possibility to write into this stream object in the future. The control will release the stream object after document closing. For possible named values, see the Booleans. Default value is **0** (while opening from stream, all document's content will be placed into temporary file at first, after opening stream object will be released immediately). *Change this property before opening any of the documents.*
RestoreLastPageDisp	Integer/String [RWS]	Specifies restoring latest page display settings (pages zoom, pages layout, view position) from corresponding recent item when opening. For possible named values, see the Booleans. Default value is **1**.
RestoreLastView	Integer/String [RWS]	Specifies restoring latest document's view (panes layout, visibility) from corresponding recent item when opening. For possible named values, see the Booleans. Default value is **0**.
SaveMethod	Integer/String [RWS]	Specifies the method which will be used for saving any documents by default. For possible named values, see the Document Save Methods. Default value is **0** (full save or incremental save for digitally signed documents).
SaveAsDestType	Integer/String [RWS]	Specifies what default folder will be used in Save As dialog. For possible named values, see the Document SaveAs Destination Types. Default value is **0** (Open last used directory).
SaveAsDestLast	Integer/String	Contain path to directory of last saved document.

	[RS]	
SaveAsDestCustom	Integer/String [RWS]	Specifies custom default directory for Save As dialog. Works only when **"SaveAsDestType"** is set to **"Custom"**.
AllowSaveCustomProducer	Integer/String [RWS]	Allows to save custom Producer name into a document when saving. For possible named values, see the Booleans. Default value is **0**(The default producer name will be saved, like as "PDF-XChange...").
DownloadFolder	String [RWS]	The folder for all downloaded documents. Used by "Open from URL" feature.

Remarks

This object is an array of items, where each item is represented as <Item>. For more information about object naming notation, see Object Name Notation.

Examples for usage (in pseudocode):

1.1. get documents count:

```
...
DoVerb("Documents.Count", "get", DataIn, DataOut, 0);
// or:
GetProperty("Documents.Count", DataOut, 0);
...
```

1.2. get unique ID of active document:

```
...
DoVerb("Documents.Active", "get", DataIn, DataOut, 0);
// or:
GetProperty("Documents.Active", DataOut, 0);
// or:
GetActiveDocument(ID);
...
```

1.3. get document unique ID by order index (0):

```
...
DoVerb("Documents[0].ID", "get", DataIn, DataOut, 0);
// or:
GetProperty("Documents[0].ID", DataOut, 0);
// or:
GetDocumentID(0, ID);
...
```

1.4. get document file name by unique ID (4095):

```
...
DoVerb("Documents[#4095].FileName", "get", DataIn, DataOut, 0);
// or:
GetProperty("Documents[#4095].FileName", DataOut, 0);
// or:
GetDocumentProperty(4095, "FileName", DataOut, 0);
...
```

1.5. get document file name by order index (0):

```
...
DoVerb("Documents[0].FileName", "get", DataIn, DataOut, 0);
// or:
GetProperty("Documents[0].FileName", DataOut, 0);
...
```

1.6. get document title:

```
...
DoVerb("Documents[#4095].Title", "get", DataIn, DataOut, 0);
// or:
DoVerb("Documents["C:\Test.pdf"].Title", "get", DataIn, DataOut, 0);
// or:
```

```
GetProperty("Documents[#4095].Title", DataOut, 0);
// or:
GetDocumentProperty(4095, "Title", DataOut, 0);
...
```

2.1. set active document:

```
...
// use simple operation:
DoVerb(NULL, "ActivateDocument", DataIn(4095), DataOut, 0);
// or use property-object:
DoVerb("Documents.Active", "set", DataIn(4095), DataOut, 0);
// or:
SetProperty("Documents.Active", DataIn(4095), 0);
// or use method of interface:
ActivateDocument(4095);
...
```

2.2. set document title:

```
...
DoVerb("Documents[#4095].Title", "set", DataIn("Sample Title"), DataOut, 0
// or:
SetProperty("Documents[#4095].Title", DataIn("Sample Title"), 0);
// or:
SetDocumentProperty(4095, "Title", DataIn("Sample Title"), 0);
...
```

2.3. close document:

```
...
// use simple operation:
DoVerb(NULL, "CloseDocument", DataIn(4095), DataOut, 0);
// or use object-depended operation:
DoVerb("Documents[#4095]", "close", DataIn, DataOut, 0);
// or:
DoDocumentVerb(4095, NULL, "close", DataIn, DataOut, 0);
// or use method of interface:
CloseDocument(4095);
...
```

See Also

IPDFXCview::DoVerb, IPDFXCview::DoDocumentVerb,
IPDFXCview::GetProperty, IPDFXCview::SetProperty,
IPDFXCview::GetDocumentProperty, IPDFXCview::SetDocumentProperty,
IPDFXCview::SaveDocument

2.2.2.4.1 <Item>

Defines item template for each opened document representation.

Contained Objects

Name	Type	Description
ID	Integer [R]	Defines the unique *identifier* of the document. This is not the same as the document *index*, which may change as documents are opened or closed.
Name	String [R]	Contains full file name of the document. Defines the unique string *identifier* of the document.
FileName	String [R]	Same as **Name**.
DispFileName	String [RW]	Specifies any custom name for displaying in the UI. Can be empty.
Title*	String [RW]	Document title.
Subject*	String	Document subject.

	[RW]	
Author*	String [RW]	Document author.
Keywords*	String [RW]	Document keywords.
Producer*	String [RW]	Document producer.
Creator*	String [RW]	Document creator.
CreationDate*	String [RW]	Document creation date.
ModificationDate*	String [RW]	Document modification date.
Version*	String [RW]	Document PDF-version.
Modified	Integer/ String [RW]	Document modification flag. Check this value for detection if the document is modified or set this value to **0** to discard changes on closing or set to **1** to effect the document modification. For possible named values, see Booleans.
NoClose	Integer/ String [RW]	Specifies restrict for closing PDF document. For possible named values, see Booleans. Default value is **0** (false).
View	Group [RW]	Controls document view attributes: visibility, positions, layouts for bars, panes, windows. It also provides customizing of toolbars.
Pages	Array [RW]	Represents an array of all existing pages of the document and provides additional operations with them.
Form	Group [RW]	Allows to specify PDF form settings for the document.
ReadOnly	Integer/ String [RW]	The purpose of this feature is to forbid the user to change the document. Default value is **0** (allow user to change the document). For possible named values, see Booleans.
ContentReady	Integer/ String [R]	Flag that signals about the completeness of displaying the content of all visible pages. This flag depends on Objects::Documents. ContentMonitor property. If **Content Monitor** is enabled then value of this property can be changed asynchronously.Notification about each change of this property can be received through _IPDFXCviewEvents::OnEvent. For possible named values, see Booleans.
Temp	Integer/ String [RW]	Marks the document as temporary. For end-user - if it is **1** then "Save" operation displays the "Save As" dialog and sets it to **0** automatically, when save operation was succeeded. This flag will be ignored if save operation is called by developer. For possible named values, see Booleans. Default value is **0**.
PDFA	Integer/	If it means **1**(true) then document has **PDF/A**

	String [R]	specific. For possible named values, see Booleans.
DenyMovePages	Integer/ String [RW]	Forbids the moving pages inside document or outside. For possible named values, see Booleans. Default value is **0**.
DenyInsertPages	Integer/ String [RW]	Forbids the inserting new pages into the document. For possible named values, see Booleans. Default value is **0**.
DenyDeletePages	Integer/ String [RW]	Forbids the deleting of existing pages from the document. For possible named values, see Booleans. Default value is **0**.
GetSelMarkups	Integer/ String [RW]	Controls the behavior of Objects::Documents::<Item>::GetSelectedAnnot function.

*****Per Adobe's PDF Reference, each document may contain certain *info tags.* Some of these tags are represented by marked properties above.

Contained Methods

Name	Description
AddAttachment	Attaches a file to the document.
ClearSelection	Clears selection for document.
Close	Closes document.
DeletePages	Delete pages from PDF document.
Export	Exports document to an other formats.
ExtractPages	Extracts pages from PDF document.
Flush	Flush document to an other formats.
GetAllSelectedText	Get all selected text from document.
GetAllText	Get all text from document.
GetSelectedPageThumbnails	Gets selected page thumbnails in the thumbnails view of the document.
GetSelectionState	Get selection state for document.
HighlightSelection	Highlights entire selection.
InsertEmptyPages	Inserts empty pages to PDF document.
InsertPages	Inserts pages from source document into current document.
IsOperationGranted	Returns the permission for operation with the document.
Print	Prints document.
RotatePages	Rotates pages in document.
Save	Saves document.
SelectAllText	Selects all text in document.
SummarizeAnnots	Summarizes all comments from a document.
GetSelectedAnnot	Gets identification info of the selected annotation.
GetSelectedField	Gets full name of the selected form field.
GetSelectedWidget	Gets identification info of the selected widget.
SelectAnnot	Selects an annotation on the page and sets input focus for it.
SelectField	Selects an field (widget) on the page and sets input focus for it.

GetAttachmentsCount	Gets count of the document's attachments.
GetAttachmentName	Gets name of the attachment.
GetAttachmentDesc	Gets description of the attachment.
GetAttachmentSize	Gets size of the attachment.
GetAttachmentModDate	Gets the last modification date of the attachment.
SaveAttachment	Saves document's attachment to a file or to an stream object.
DeleteAttachment	Removes attachment from the document.
DeleteAllAttachments	Removes all attachments from the document.

See Also

IPDFXCview::DoDocumentVerb, Objects::Documents

2.2.2.4.1.1 <Methods>

See Objects::Documents::<Item> Methods.

Attaches a file to the document.

Name

```
AddAttachment
```

Arguments

#	Req.	Type	Description
1	Yes	BSTR/IStream	The source: local file name, URL, or an stream object.
2	No	BSTR	Name of the new attachment, will be displayed in the **Attachments Pane**.
3	No	BSTR	Description of the new attachment, will be displayed in the **Attachments Pane**.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DoDocumentVerb(docId, NULL, "AddAttachment",
       DataIn("C:\TestCopy.pdf", "MyAttachment", "This is my attachment"),
...
```

See Also

IPDFXCview::DoVerb, Objects::Documents::<Item>, IPDFXCargs

Clears the operations history for the document. The special operations history is used for Undo/Redo feature.

Name

```
ClearOperationsHistory
```

Arguments

```
None
```

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DoDocumentVerb(docID, NULL, "ClearOperationsHistory", NULL, NULL, 0);
...
```

See Also

IPDFXCview::DoDocumentVerb

Clears selection for current document.

Name

```
ClearSelection
```

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// Clears selection for first document:
DoVerb("Documents[0]", "ClearSelection", NULL, NULL, 0);
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>

Closes the document.

Name

```
Close
```

Arguments

```
None
```

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// Close the first document:
DoVerb("Documents[0]", "Close", NULL, NULL, 0);
...
```

See Also

IPDFXCview::DoVerb, IPDFXCview::CloseDocument, Objects::Documents::<Item>

Removes all attachments from the document.

Name

```
DeleteAllAttachments
```

Arguments

```
None
```

Return Value

Returns **S_OK** or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoDocumentVerb

Removes the attachment from the document.

Name

```
DeleteAttachment
```

Arguments

#	Req.	Type	Description
1	Yes	LONG/ BSTR	The zero-based index of the attachment in the attachments list **or** name of the attachment.

Return Value

Returns **S_OK** or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoDocumentVerb

Delete pages from PDF document using parameters previously defined by Objects::Operations.DeletePages.

Name

```
DeletePages
```

Arguments

```
None
```

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

1. Show delete pages dialog:

```
...
DoVerb("Documents[0]", "DeletePages", NULL, NULL, 0);
...
```

2. Delete 1-5,10 pages from current PDF document:

```
...
SetProperty("Operations.DeletePages.RangeType", "Exact");
SetProperty("Operations.DeletePages.RangeText", "1-5,10"); // pages
...
DoVerb("Documents[0]", "DeletePages", NULL, NULL, PXCVA_NoUI);
...
```

See Also

IPDFXCview::DoVerb,
Operations::DeleteDocumentPages,
Objects::Operations.DeletePages

Exports a document to an image file(s) with the format parameters previously defined by Objects::Export.

Name

```
Export
```

Arguments

```
None
```

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// Print the first document:
DoVerb("Documents[0]", "Export", NULL, NULL, 0);
...
```

See Also

IPDFXCview::DoVerb,
IPDFXCview::ExportDocument,
Objects::Documents::<Item>

Extracts pages from PDF document using parameters previously defined by Objects::Operations.ExtractPages.

Name

```
ExtractPages
```

Arguments

```
None
```

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

1. Show extract pages dialog:

```
...
DoVerb("Documents[0]", "ExtractPages", NULL, NULL, 0);
...
```

2. Extract 1-5,10 pages from current PDF document:

```
...
SetProperty("Operations.ExtractPages.RangeType", "Exact");
SetProperty("Operations.ExtractPages.RangeText", "1-5,10"); // pages
...
DoVerb("Documents[0]", "ExtractPages", NULL, NULL, PXCVA_NoUI);
...
```

See Also

IPDFXCview::DoVerb,
Operations::ExtractDocumentPages,
Objects::Operations.ExtractPages

Flushes all of the user's cached changes for a document, stops editing and updates document's structure. Can be used before saving of the document.

Name

```
Flush
```

Arguments

```
None
```

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoVerb, IPDFXCview::FlushDocument, Objects::Documents::<Item>

Gets all selected text from current document.

Name

```
GetAllSelectedText
```

Arguments

#	Req.	Type	Description
1	No	BSTR	Specifies path to output file in which text will be stored.
		Stream	Specifies stream object in which text will be stored.
2	No	Boolean	Exclude new line characters from output (CR\LF). Default value is **false**.

Outputs

#	Type	Description
1	BSTR	Output text. Appears only when first argument is not specified.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// Gets all text to string variable (DataOut):
DoVerb("Documents[0]", "GetAllSelectedText", NULL, DataOut, 0);
// Gets all text to specified file:
DoVerb("Documents[0]", "GetAllSelectedText", "C:\PdfText.txt", NULL, 0);
// Gets all text to stream object which contained in DataIn:
DoVerb("Documents[0]", "GetAllSelectedText", DataIn, NULL, 0);
...
```

See Also

IPDFXCview::DoVerb, Objects::Documents::<Item>

Gets all text from current document.

Name

```
GetAllText
```

Arguments

#	Req.	Type	Description
1	No	BSTR	Specifies path to output file in which text will be stored.
		Stream	Specifies stream object in which text will be stored.
2	No	Boolean	Exclude new line characters from output (CR\LF). Default value is **false**.

Outputs

#	Type	Description
1	BSTR	Output text. Appears only when first argument is not specified.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// Gets all text to string variable (DataOut):
DoVerb("Documents[0]", "GetAllText", NULL, DataOut, 0);
// Gets all text to specified file:
DoVerb("Documents[0]", "GetAllText", "C:\PdfText.txt", NULL, 0);
// Gets all text to stream object which contained in DataIn:
DoVerb("Documents[0]", "GetAllText", DataIn, NULL, 0);
...
```

See Also

IPDFXCview::DoVerb, Objects::Documents::<Item>

Gets name of the attachment.

Name

```
GetAttachmentName
```

Arguments

#	Req.	Type	Description
1	Yes	LONG/ BSTR	The zero-based index of the attachment in the attachments list **or** name of the attachment.

Outputs

#	Type	Description
1	BSTR	The description of the attachment.

Return Value

Returns **S_OK** or an error value otherwise. To obtain the text description of a received error

code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoDocumentVerb

Gets the last modification date of the attachment.

Name

```
GetAttachmentModDate
```

Arguments

#	Req.	Type	Description
1	Yes	LONG/ BSTR	The zero-based index of the attachment in the attachments list **or** name of the attachment.

Outputs

#	Type	Description
1	LONGLONG (INT64)	Contains a 64-bit value representing the number of 100-nanosecond intervals since January 1, 1601 (UTC), same as in standard windows **FILETIME** structure.

Return Value

Returns **S_OK** or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoDocumentVerb

Gets name of the attachment.

Name

```
GetAttachmentName
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The zero-based index of the attachment in the list of attachments.

Outputs

#	Type	Description
1	BSTR	The name of the attachment.

Return Value

Returns **S_OK** or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoDocumentVerb

Gets count of the document's attachments.

Name

```
GetAttachmentsCount
```

Arguments

```
None
```

Outputs

#	Type	Description
1	LONG	The count of attachments.

Return Value

Returns **S_OK** or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoDocumentVerb

Gets name of the attachment.

Name

```
GetAttachmentSize
```

Arguments

#	Req.	Type	Description
1	Yes	LONG/ BSTR	The zero-based index of the attachment in the attachments list **or** name of the attachment.

Outputs

#	Type	Description
1	LONGLONG (INT64)	The uncompressed size of the attachment, in bytes.

Return Value

Returns **S_OK** or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoDocumentVerb

Gets identification info of the selected annotation.

Name

```
GetSelectedAnnot
```

Arguments

```
None
```

Outputs

#	Type	Description
1	LONG	The zero-based index of page with the selected annotation. But, if no selected annotation then control will return **-1** as page index and result of method calling will be **S_FALSE**.
2	LONG	The zero-based index of annotation on the page.

Return Value

Returns **S_OK** if is selected annotation, **S_FALSE** instead, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DoDocumentVerb(docID, NULL, "GetSelectedAnnot", NULL, DataOut, 0);
int pageIndex = DataOut[0];
if (pageIndex >= 0) // we have selected (focused) annotation
{
   indexOnPage = DataOut[1];
...
```

Note: exists special property of document which can some change behavior of this function:

```
...
SetSocumentProperty(docID, "GetSelMarkupsOnly", "true");
...
// next code will return selection info only if selected annotation is mar
// this behaviour is same as in Java-Script function Doc::getAnnots()
DoDocumentVerb(docID, NULL, "GetSelectedAnnot", NULL, DataOut, 0);
...
```

See Also

IPDFXCview::DoDocumentVerb, Objects::Documents::<Item>::SelectAnnot

Gets full name of the selected form field. The form field in PDF is not a visual object, but each form field in the document may have one or more corresponding visual objects on page(s) which are called **widgets**. Widget is a visual object for form field: text box for text field, check-button for "on/off" field, etc. So, each form field can have a list of widgets which can be located at different places of page, or on different pages, may have different visual attributes (colors, borders, fonts etc.)

Name

```
GetSelectedField
```

Arguments

```
None
```

Outputs

#	Type	Description
1	BSTR	The full name of selected (with input focus) form field (widget). If no selected field the control will return empty string and result of method calling will be **S_FALSE**.

Return Value

Returns **S_OK** if is selected form field, **S_FALSE** instead, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DoDocumentVerb(docID, NULL, "GetSelectedField", NULL, DataOut, 0);
string fieldName = DataOut;
if (fieldName == "Form1.Group1.TextBox1.0")
{ }
...
```

To obtain the selected widget exactly use the additional Objects::Documents::<Item>::GetSelectedWidget.

See Also

IPDFXCview::DoDocumentVerb, Objects::Documents::<Item>::SelectField

Gets selected page thumbnails in the thumbnails view of the document.

Name

```
GetSelectedPageThumbnails
```

Outputs

Can contains one or more integer numbers which represents the indexes (0-based) of the selected page thumbnails in the thumbnails view of the document. These numbers will be obtained through **SAFEARRAY**.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DoDocumentVerb(docId, "GetSelectedPageThumbnails", DataIn, DataOut, 0);
int firstPageThumbSel = DataOut[0];
int secondPageThumbSel = DataOut[1];
```

For simplify in/out operations with **SAFEARRAY** you can use the IPDFXCargs object:

```
...
// obtain a new IPDFXCargs object with array of numbers:
```

```
DoDocumentVerb(docId, "GetSelectedPageThumbnails", DataIn, DataOut, PXCVA_
IPDFXCargsObj args = DataOut;
int firstPageThumbSel = args[0];
int secondPageThumbSel = args[1];
...
```

See Also

IPDFXCview::DoVerb, IPDFXCview::DoDocumentVerb

Gets identification info of the selected widget (visual object for corresponding form field).

Name

```
GetSelectedWidget
```

Arguments

```
None
```

Outputs

#	Type	Description
1	LONG	The zero-based index of page with the selected widget. But, if no selected widget then control will return **-1** as page index and result of method calling will be **S_FALSE**.
2	LONG	The zero-based index of widget on the page.

Return Value

Returns **S_OK** if is selected widget, **S_FALSE** instead, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DoDocumentVerb(docID, NULL, "GetSelectedWidget", NULL, DataOut, 0);
int wgPageIndex = DataOut[0];
if (wgPageIndex >= 0) // we have selected (focused) widget
{
   wgIndexOnPage = DataOut[1];
```

See Also

IPDFXCview::DoDocumentVerb, Objects::Documents::<Item>::GetSelectedField

Gets text selection state for the document.

Name

```
GetSelectionState
```

Outputs

#	Type	Description
1	Boolean	if false (**0**) then document doesn't have any selection.

		if true (**1**) then some text in document is selected. Selected text may be obtain by using GetAllSelectedText method.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// Gets selection state for the first opened document:
DoVerb("Documents[0]", "GetSelectionState", DataIn, DataOut, 0);
....
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>,
Objects::Documents::<Item>.GetAllSelectedText

Highlights selection.

Name

```
HighlightSelection
```

Arguments

#	Req.	Type	Description
1	No	LONG	The color (COLORREF) of the highlight. By default means **#00FFFF** (yellow).
2	No	Double	The opacity of the highlight comment. By default set to **1.0** (100% opaque).

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// Highlight selection with violet color and 65% opacity
DoVerb("Documents[0]", "HighlightSelection", DataIn(0x800080, 0.65), NULL,
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>

Highlights text in the document by special input file with information for highlighting of the text. See adobe's Highlight File Format.
The short description of this format (pseudo xml):

```
<XML>
```

```
<Body units=[characters|words] version=verNum type=[highlight|underline|strikeout]
color=[rgb(r,g,b)|#FFFFFFFF] opacity=[0..1]>
<Highlight>
<loc pg=pageIndex pos=startChar len=charsCount type=[highlight|underline|strikeout]
color=[rgb(r,g,b)|#FFFFFFFF] opacity=[0..1]>
...
</Highlight>
</Body>
</XML>
```

Note: the blue parameters are unsupported by adobe, but are supported by our products (this ActiveX, browser plugins, standalone PDF-Viewer).

Name

```
HighlightTextByFile
```

Arguments

#	Req.	Type	Description
1	Yes	BSTR/IStream	The local file name, URL, or an stream object with information for text highlighting.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DoDocumentVerb(docId, NULL, "HighlightTextByFile",
       DataIn("C:\HighlightText.xml"), NULL);
...
```

See Also

IPDFXCview::DoDocumentVerb

Inserts empty pages to PDF document using parameters previously defined by Objects::Operations.InsertEmptyPages.

Name

```
InsertEmptyPages
```

Arguments

```
None
```

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

1. Show Insert Empty pages dialog:

```
...
DoVerb("Documents[0]", "InsertEmptyPages", NULL, NULL, 0);
...
```

2. Insert 5 empty A4 pages to the begin of current PDF document:

```
...
SetProperty("Operations.InsertEmptyPages.Count", 5);
SetProperty("Operations.InsertEmptyPages.InsertBefore", 0);
SetProperty("Operations.InsertEmptyPages.PaperMode", "Standard");
SetProperty("Operations.InsertEmptyPages.PaperName", "A4");
...
DoVerb("Documents[0]", "InsertEmptyPages", NULL, NULL, PXCVA_NoUI);
...
```

See Also

IPDFXCview::DoVerb,
Operations::InsertEmptyDocumentPages,
Objects::Operations.InsertEmptyPages

Inserts pages from source document into current document using parameters previously defined by Objects::Operations.InsertPages.

Name

```
InsertPages
```

Arguments

```
None
```

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

1. Show Insert Pages dialog:

```
...
DoVerb("Documents[0]", "InsertPages", NULL, NULL, 0);
...
```

2. Insert 1, 3 pages from PDF file to the begin of current document:

```
...
SetProperty("Operations.InsertPages.RangeType", "Exact");
SetProperty("Operations.InsertPages.RangeText", "1, 3"); // pages
SetProperty("Operations.InsertPages.FromExternal", 1);
SetProperty("Operations.InsertPages.File", "c:\SomeFile.pdf");
SetProperty("Operations.InsertPages.InsertBefore", 0);
...
DoVerb("Documents[0]", "InsertPages", NULL, NULL, PXCVA_NoUI);
...
```

See Also

IPDFXCview::DoVerb,
Operations::InsertDocumentPages,
Objects::Operations.InsertPages

Returns the permission for operation with the document.

Name

```
IsOperationGranted
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	Specifies the **object identifier**.
2	Yes	LONG	Specifies the **operation identifier** which is depended with the object.

Object Identifiers

Value	Description
1	Document
2	Page
3	Link
4	Bookmark
5	Thumbnail
6	Annotation
7	Form
8	Signature
9	Named Embedded File

Operation Identifiers

Value	Description
1	Used for checking to see if all operations are allowed
2	Create
3	Delete
4	Modify
5	Copy
6	Use Accessibility
7	Select
8	Open (document)
9	Change Security Settings (for document)
10	High Quality Printing (for document)
11	Low Quality Printing (for document)
12	Form fill-in, or sign the existing field
13	Rotate (pages)
14	Crop (pages)

15	Summarize (annotations)
16	Insert (pages)
17	Replace (pages)
18	Reorder (pages)
19	Full Save (document)
20	Import (form-data, annotations)
21	Export (form-data, annotations, images)
22	Used for checking to see if any operation is allowed
24	Submit forms outside of the browser
25	Allow the form to spawn template pages

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

1. Check if is possible to insert pages:
```
...
DoDocumentVerb(docId, NULL, "IsOperationGranted", dataIn(2,16), NULL, 0);
...
```
2. Check if is possible to hight quality print:
```
...
DoDocumentVerb(docId, NULL, "IsOperationGranted", dataIn(1,10), NULL, 0);
...
```

See Also

IPDFXCview::DoDocumentVerb

Prints document pages using parameters previously defined by Objects::Print.

Name

```
Print
```

Arguments

```
None
```

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// Displays print dialog for first document:
DoVerb("Documents[0]", "Print", NULL, NULL, 0);
// Print first document directly, without dialog:
DoVerb("Documents[0]", "Print", NULL, NULL, PXCVA_NoUI);
```

```
...
```

See Also

IPDFXCview::DoVerb,
IPDFXCview::PrintDocument,
Objects::Documents::<Item>

Removes the document protection. Can be successful only if the document has been opened with owner permissions.
For example, if the document is the password-protected then you should specify the special *owner password* for opening.

Name

```
RemoveSecurity
```

Arguments

```
None
```

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DoDocumentVerb(docId, "", "RemoveSecurity", NULL, NULL, 0);
...
```

See Also

IPDFXCview::DoVerb, IPDFXCview::CloseDocument,
Objects::Documents::<Item>

Rotates pages in document using parameters previously defined by Objects::Operations.RotatePages.

Name

```
RotatePages
```

Arguments

```
None
```

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

1. Show rotate pages dialog:

```
...
DoVerb("Documents[0]", "RotatePages", NULL, NULL, 0);
...
```

2. Rotates 1-5,10 pages to 90 degrees in current PDF document:

```
...
SetProperty("Operations.RotatePages.RangeType", "Exact");
SetProperty("Operations.RotatePages.RangeText", "1-5,10"); // pages
SetProperty("Operations.RotatePages.Direction", "Clockwise");
...
DoVerb("Documents[0]", "RotatePages", NULL, NULL, PXCVA_NoUI);
...
```

See Also

IPDFXCview::DoVerb,
Operations::RotateDocumentPages,
Objects::Operations.RotatePages

Saves a document in a number of different ways depending on the **Flags** parameter in **DoVerb** function.

Name

```
Save
```

Arguments

#	Req.	Type	Description
1	No	BSTR/IStream	The destination file name, or Stream Object.
2	No	LONG	The save flags. For more information, see PXCVA_DocumentSaveFlags.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

Use this operation to save or copy the specified document. To display the standard Save File dialog, pass NULL as the Destination file name. If you want to skip possible error dialogs then you must set the PXCVA_Flags::PXCVA_NoUI flag in the **Flags** argument.

For example (in pseudocode):

```
...
// save first document directly to "C:\TestCopy.pdf" file:
DoVerb("Documents[0]", "Save", DataIn("C:\TestCopy.pdf"), NULL, PXCVA_NoUI
// save first document to Stream:
DoVerb("Documents[0]", "SaveDocument", DataIn(StreamObj), NULL, PXCVA_NoUI
...
```

The Stream Object can be asynchronous.

See Also

IPDFXCview::DoVerb, IPDFXCview::SaveDocument,
Objects::Notifications::BeforeSaveDoc, Objects::Notifications::DocSaved,
Objects::Documents::UseStreamsDirectly

Saves document's attachment to a file or to stream object.

Name

```
SaveAttachment
```

Arguments

#	Req.	Type	Description
1	Yes	LONG /BSTR	The zero-based index of the attachment in the attachments list **or** name of the attachment.
2	No	BSTR/ IStrea m	The destination: local file name or stream object. if isn't specified or is NULL then control will display the UI for select destination.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

See Also

IPDFXCview::DoDocumentVerb, IPDFXCargs

Selects text in entire document.

Name

```
SelectAllText
```

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// Selects all text in the first document:
DoVerb("Documents[0]", "SelectAllText", NULL, NULL, 0);
...
```

See Also

IPDFXCview::DoVerb, Objects::Documents::<Item>

Selects an annotation on the page and sets input focus for it.

Name

```
SelectAnnot
```

Arguments

#	Req.	Type	Description

1	Yes	BSTR	The annotation name. In this case the control will lookup all annotations in document for the specified name and stop on first match.

or

#	Req.	Type	Description
1	Yes	LONG	The zero-based index of page with the annotation.
2	Yes	LONG	The zero-based index of annotation object on the page.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

Simple examples (in pseudocode):

```
...
// select annotation by name:
DoDocumentVerb(docId, NULL, "SelectAnnot", DataIn("myFineComment"), NULL,
// select annotation by indexes:
DoDocumentVerb(docId, NULL, "SelectAnnot", DataIn(pageIndex, indexOnPage),
               NULL, PXCVA_NoUI);
...
```

See Also

IPDFXCview::DoDocumentVerb, Objects::Documents::<Item>::GetSelectedAnnot

Selects a form field on the page and sets input focus for it. The form field in PDF is not a visual object, but each form field in the document may have one or more corresponding visual objects on page (s) which are called **widgets**. Widget is a visual object for form field: text box for text field, check-button for "on/off" field, etc. So, each form field can have a list of widgets which can be located at different places of page, or on different pages, may have different visual attributes (colors, borders, fonts etc.)

Name

```
SelectField
```

Arguments

#	Req.	Type	Description
1	Yes	BSTR	The full form field name. Examples: "TextBox1", "TextBox1.0", "TextBox1.1". The suffixes ".0", ".1", means the zero-based index of the corresponding **widget** in the widgets list of the form field. If widget isn't specified then first widget of the field will be selected automatically.

or to select widget exactly

#	Req.	Type	Description
1	Yes	LONG	The zero-based index of page with the widget.
2	Yes	LONG	The unique, zero-based index of the widget object on the page.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

Simple examples (in pseudocode):

```
...
// select field(widget) by name:
DoDocumentVerb(docId, NULL, "SelectField", DataIn("Text1"), NULL, PXCVA_Nc
DoDocumentVerb(docId, NULL, "SelectField", DataIn("Text1.0"), NULL, PXCVA_
DoDocumentVerb(docId, NULL, "SelectField", DataIn("Text1.2"), NULL, PXCVA_
// select field by widget indexes:
DoDocumentVerb(docId, NULL, "SelectField", DataIn(wgPageIndex, wgIndexOnPa
               NULL, PXCVA_NoUI);
// or identical:
DoDocumentVerb(docId, NULL, "SelectAnnot", DataIn(wgPageIndex, wgIndexOnPa
               NULL, PXCVA_NoUI);
...
```

See Also

IPDFXCview::DoDocumentVerb, Objects::Documents::<Item>::GetSelectedField

Summarizes all comments from a document to a new document(file) using parameters previously defined by Objects::Operations.SummarizeAnnots.

Name

```
SummarizeAnnots
```

Arguments

```
None
```

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

1. Show summarize annotations dialog:

```
...
DoVerb("Documents[0]", "SummarizeAnnots", NULL, NULL, 0);
...
```

2. Save all comments from 1-5,10 pages to text file:

```
...
SetProperty("Operations.SummarizeAnnots.Output.Type", "txt");
SetProperty("Operations.SummarizeAnnots.RangeType", "Exact");
SetProperty("Operations.SummarizeAnnots.RangeText", "1-5,10"); // pages
SetProperty("Operations.SummarizeAnnots.Output.AutoView", "false");
SetProperty("Operations.SummarizeAnnots.Output.TXT.CodePage", "UTF8");
SetProperty("Operations.SummarizeAnnots.Output.TXT.FolderName", "C:\");
SetProperty("Operations.SummarizeAnnots.Output.TXT.FileName",
"All Comments From <File Name>");
...
DoVerb("Documents[0]", "SummarizeAnnots", NULL, NULL, PXCVA_NoUI);
...
```

See Also

IPDFXCview::DoVerb,
Operations::SummarizeDocumentAnnots,
Objects::Operations.SummarizeAnnots

2.2.2.4.1.2 Pages

This object represents an array of all pages of the document and provides additional operations with them.

Item Template

<Item> - defines item template for each page representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains number of all pages in the document.
Current	Integer [RW]	Contains index of the current page.
Zoom	Double/ String [RW]	Specifies page magnification coefficient (zoom) in the page view window. For possible values, see the Pages Magnifications.
ZoomMode	Integer/ String [R]	Specifies page magnification mode. For possible values, see the Pages Magnification Modes.
Layout	Integer/ String [RW]	Specifies page layout in the page view window. For possible values, see the Pages Layouts.
Width	Integer/ String [R]	Contains actual width in pixels of the pages view.
Height	Integer/ String [R]	Contains actual height in pixels of the pages view.
ScreenX	Integer/ String [R]	Contains actual X-position of the top-left corner of pages view window, relative to desktop.
ScreenY	Integer/ String [R]	Contains actual Y-position of the top-left corner of pages view window, relative to desktop.
DisplayRotation	Integer/ String [RW]	Specifies the display rotation factor of the page view - number of degrees, multiple 90.

Contained Methods

Name	Description
GetSelectedRanges	Gets pages with any selection.

Remarks

This object is an array of items, each item is represented by <Item>. For more information about object names notation, see Object Name Notation.

Examples for usage (in pseudocode):

1.1. get pages count:

```
...
DoVerb("Documents[#4095].Pages.Count", "get",
      DataIn, DataOut, 0);
// or:
GetProperty("Documents[#4095].Pages.Count", DataOut, 0);
// or:
DoDocumentVerb(4095, "Pages.Count", "get",
       DataIn, DataOut, 0);
// or:
GetDocumentProperty(#4095, "Pages.Count", DataOut, 0);
...
```

1.2. get width of first page:

```
...
DoVerb("Documents[#4095].Pages[0].Width", "get",
       DataIn, DataOut, 0);
// or:
GetProperty("Documents[#4095].Pages[0].Width", DataOut, 0);
// or:
GetDocumentProperty(4095, "Pages[0].Width", DataOut, 0);
...
```

2.1. set pages layout:

```
...
// set 'continuous' layout:
DoVerb("Documents[#4095].Pages.Layout", "set",
       DataIn(1), DataOut, 0);
DoDocumentVerb(4095, "Pages.Layout", "set",
       DataIn(1), DataOut, 0);
// or put named value:
DoVerb("Documents[#4095].Pages.Layout", "set",
       DataIn("Continuous"), DataOut, 0);
// or:
SetProperty("Documents[#4095].Pages.Layout",
       DataIn("Continuous"), 0);
// or:
SetDocumentProperty(4095, "Pages.Layout",
       DataIn("Continuous"), 0);
...
```

2.2. change width of first page:

```
...
DoVerb("Documents[#4095].Pages[0].Width", "set",
       DataIn(100.0), DataOut, 0);
// or:
SetProperty("Documents[#4095].Pages[0].Width",
        DataIn(100.0), 0);
// or:
SetDocumentProperty(4095, "Pages[0].Width",
        DataIn(100.0), 0);
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents,
Objects::Documents::<Item>

See Objects::Documents::<Item>.Pages Methods.

Gets pages with any selection.

Name

```
GetSelectedRanges
```

Outputs

Can contains one or more pairs of numbers in a format: the first number is an index(0-based) of the first page in the selection range, the second number is count of the page in the selection range. These numbers will be obtained through **SAFEARRAY**.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DoDocumentVerb(docId, "Pages", "GetSelectedRanges", DataIn, DataOut, 0);
...
// we have selection which contains two blocks:
int firstPageWithSel1 = DataOut[0];
int countPagesWithSel1 = DataOut[1];
int firstPageWithSel2 = DataOut[2];
int countPagesWithSel2 = DataOut[3];
```

For simplify in/out operations with **SAFEARRAY** you can use the IPDFXCargs object:

```
...
// obtain a new IPDFXCargs object with array of numbers:
DoDocumentVerb(docId, "Pages", "GetSelectedRanges", DataIn, DataOut, PXCVA
IPDFXCargsObj args = DataOut;
int firstPageWithSel1 = args[0];
int countPagesWithSel1 = args[1];
int firstPageWithSel2 = args[2];
int countPagesWithSel2 = args[3];
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages

Defines item template for each opened document representation.

Contained Objects

Name	Type	Description
Width	Double [R]	Width of page in points (1 point is 1/72 inch). This value shows the width of rectangle which is calculated as intersection of **Media** and **Crop** boxes with applying of **Rotation** and **UserUnit** values (see Adobe's PDF Reference).
Height	Double [R]	Height of page in points (1 point is 1/72 inch). This value shows the height of rectangle which is calculated as intersection of **Media** and **Crop** boxes with applying of **Rotation** and **UserUnit** values (see Adobe's PDF Reference).

Rotation	Integer [R]	Represents rotation angle of the page.
MediaBox	Group [R]	Represents the **Media** box of the page. Has the Objects::RectangleF structure.
CropBox	Group [R]	Represents the **Crop** box of the page. Has the Objects::RectangleF structure.
Text	Group [R]	

Contained Methods

Name	Description
GetAnnotName	Gets name of the annotation object specified by index.
GetAnnotRect	Gets rectangle of the annotation object specified by index.
GetAnnotsCount	Gets count of annotations on the page.
GetAnnotType	Gets type identifier of the annotation object.
MovePointToScreenPoint	Moves the page point to the specified screen point.
PagePointsToViewPoints	Translates the coordinates of points from page coordinate system to the pages view coordinate system.
TranslateScreenPoint	Converts the screen coordinates in pixels of a specified point on the screen to document page coordinates in points.
ViewPointsToPagePoints	Translates the coordinates of points from pages view coordinate system to the page coordinate system.

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages

See Objects::Documents::<Item>.Pages::<Item> Methods.

Gets name of the annotation object.

Name

```
GetAnnotName
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The zero-based index of the annotation object on the page.

Outputs

#	Type	Description
1	String	The name of the annotation object.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DataIn = 1; // to second annotation on the page
DoVerb("Documents[0].Pages[2]", "GetAnnotName", DataIn, DataOut, 0);
string annotName = DataOut;
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>

Gets rectangle of the annotation object on the page.

Name

```
GetAnnotRect
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The zero-based index of the annotation object on the page.

Outputs

If successful then **DataOut** will contain the **SAFEARRAY** with four numbers in order: **left, top, right, bottom -** the rectangle of the annotation on the page, in normal page coordinate system.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DataIn = 1; // to second annotation on the page
DoVerb("Documents[0].Pages[2]", "GetAnnotRect", DataIn, DataOut, 0);
double left = DataOut[0];
double top = DataOut[1];
double right = DataOut[2];
double bottom = DataOut[3];
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>

Gets count of annotations on the page.

Name

```
GetAnnotsCount
```

Arguments

```
None
```

Outputs

#	Type	Description
1	LONG	The count of annotations on the page.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DoVerb("Documents[0].Pages[2]", "GetAnnotsCount", DataIn, DataOut, 0);
int annotsCountOnThePage2 = DataOut;
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>

Gets type identifier of the annotation object.

Name

```
GetAnnotType
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The zero-based index of the annotation object on the page.

Outputs

#	Type	Description
1	LONG	The type of the annotation object.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DataIn = 1; // to second annotation on the page
DoVerb("Documents[0].Pages[2]", "GetAnnotType", DataIn, DataOut, 0);
int annotType = DataOut;
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>

Moves the page point to the specified screen point.

Name

```
MovePointToScreenPoint
```

Arguments

#	Req.	Type	Description
1	Yes	double	Specifies x coordinate of the page point in points.
2	Yes	double	Specifies y coordinate of the page point in points.
3	Yes	LONG	Specifies x coordinate of screen point, relative to desktop. See also to the: Pages.ScreenX, Pages.ScreenY.
4	Yes	LONG	Specifies y coordinate of screen point, relative to desktop.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DoVerb("Documents[0].pages[2]", "MovePointToScreenPoint",
       DataIn(px, py, sx, sy), NULL, 0);
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>

Translates the coordinates of points from page coordinate system to the pages view coordinate system.

Name

```
PagePointsToViewPoints
```

Arguments

Coordinates of input points are stored in an array of doubles where each pair specifies the X and Y coordinates of one point.
Also you can specify one last additional nonzero value to notify function that input coordinates already include the page view transformations (crop-offset, page-rotation).

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.
The **DataOut** receives **SAFEARRAY** which contain x and y coordinates of the translated input points - all output points are relative to the top-left corner of the pages view window.

Remarks

For example (in pseudocode):

```
...
p1.x = 0.0; // pt, 1/72 inch
p1.y = 0.0;
p2.x = 100.0;
p2.y = 200.0;
DoVerb("Documents[0].pages[2]",
       "PagePointsToViewPoints", DataIn(p1.x, p1.y, p2.x, p2.y), DataOut,
v1.x = DataOut[0]; // px
v1.y = DataOut[1];
v2.x = DataOut[2];
v2.y = DataOut[3];
...
double fTransformsIncludedAlready_optional = 1; // additional flag for inp
DoVerb("Documents[0].pages[2]",
       "PagePointsToViewPoints",
       DataIn(p1.x, p1.y, p2.x, p2.y, fTransformsIncludedAlready_optional)
       DataOut, 0);
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>

Converts the screen coordinates in pixels of a specified point on the screen to document page coordinates in points.

Name

```
TranslateScreenPoint
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	Specifies x coordinate of screen point in pixels, relative to desktop.
2	Yes	LONG	Specifies y coordinate of screen point in pixels, relative to desktop.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.
The **DataOut** receives **SAFEARRAY** which contain x and y coordinates on the document page in the points.

Remarks

For example (in pseudocode):

```
...
DoVerb("Documents[0].pages[2]", "TranslateScreenPoint",
        DataIn(sx, sy), DataOut, 0);

px = DataOut[0]; // Page x coordinate
py = DataOut[1]; // Page y coordinate
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>

Translates the coordinates of points from pages view coordinate system to the page coordinate system.

Name

```
ViewPointsToPagePoints
```

Arguments

Coordinates of input points are stored in an array of doubles/integers where each pair specifies the X and Y coordinates of one point.
Also you can specify one last additional nonzero value to notify function that output coordinates should include the page view transformations (crop-offset, page-rotation), if it is not specified then function will remove these transformations automatically - i.e. will translate to the normal page coordinates system.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.
The **DataOut** receives **SAFEARRAY** which contain x and y coordinates of the translated input points.

Remarks

For example (in pseudocode):

```
...
p1.x = 0.0; // pt, 1/72 inch
p1.y = 0.0;
p2.x = 100.0;
p2.y = 200.0;
DoVerb("Documents[0].pages[2]",
       "ViewPointsToPagePoints", DataIn(p1.x, p1.y, p2.x, p2.y), DataOut,
v1.x = DataOut[0]; // px
v1.y = DataOut[1];
v2.x = DataOut[2];
v2.y = DataOut[3];
...
double fIncludeTransformsToOutput_optional = 1; // additional flag for out
DoVerb("Documents[0].pages[2]",
       "ViewPointsToPagePoints",
       DataIn(p1.x, p1.y, p2.x, p2.y, fIncludeTransformsToOutput_optional)
       DataOut, 0);
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>

This object represents operations with text on a specific page of the document.

Contained Objects

Name	Type	Description
Chars	Group [R]	Represents info for characters of an array object.

Words	Group [R]	Represents info for words in an array object.
Lines	Group [R]	Represents info for text lines in an array object.

Contained Methods

Name	Description
AddLink	Place link over text range on the page.
Get	Gets text from a specified range on the page.
GetSelected	Gets selected text on the page.
GetSelectedRanges	Gets text selection ranges on the page.
GetQuads	Gets text coordinates.
Highlight	Highlights all characters from a specified range on the page.
Select	Selects all characters from a specified range on the page.
StrikeOut	Strikes out all characters from a specified range on the page.
Underline	Underlines all characters from a specified range on the page.

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>,

How to Enumerate Characters in Document?

See Objects::Documents::<Item>.Pages::<Item>.Text Methods.

Place link over text range on the page.

Name

```
AddLink
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The index of the first character.
2	Yes	LONG	The count of characters.
3	Yes	LONG /String	The target page index (zero-based) or target URL.
4	No	LONG	Specify non-zero number to make rectangular link instead control will create quadrilateral link by default.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// place web-link over first 10 characters on first page:
DoVerb("Documents[0].Pages[0].Text", "AddLink", DataIn(0, 10, "www.server.
// place go-to-page-link (to 10th page) over first 10 characters on first
DoVerb("Documents[0].Pages[0].Text", "AddLink", DataIn(0, 10, 9), NULL, 0)
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text

Gets text from a specified range on the page.

Name

```
Get
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The index of the first character of the get range.
2	Yes	LONG	The count of characters to get.
3	No	Boolean	Exclude new line characters from output (CR\LF). Default value is **false**.

Outputs

#	Type	Description
1	BSTR	Output text.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// Gets first 10 characters on the page:
DoVerb("Documents[0].Pages[0].Text", "Get", DataIn(0, 10), DataOut, 0);
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text

Gets text coordinates from a specified character range on the page.

Name

```
GetQuads
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The index of the first character of the range.
2	Yes	LONG	The count of characters in range.

Outputs

Can contains four or more pairs of numbers. These numbers will be obtained through **SAFEARRAY**. Coordinates are stored in an array of doubles where each pair specifies the X and Y coordinates of one point; each eight doubles(four points) specifies the quadrilateral area. Order of points is displayed in the Quads.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// Gets coordinates of first 10 characters on the page:
DoVerb("Documents[0].Pages[0].Text", "GetQuads", DataIn(0, 10), DataOut, 0
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text, PXCVA_OutArgs

Gets selected text on page.

Name

```
GetSelected
```

Arguments

#	Req.	Type	Description
1	No	Boolean	Exclude new line characters from output (CR\LF). Default value is **false**.

Outputs

#	Type	Description
1	BSTR	Output text.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// Gets selected text on the page (without CR\LF):
```

```
DoVerb("Documents[0].Pages[0].Text", "GetSelected", DataIn(1), DataOut, 0)
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text

Gets text selection ranges on the page.

Name

```
GetSelectedRanges
```

Outputs

Can contains one or more pairs of numbers in a format: the first number is an index(0-based) of the first character in the selection range, the second number is count of the characters in the selection range. These numbers will be obtained through **SAFEARRAY**.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
DoDocumentVerb(docId, "Pages[0].Text", "GetSelectedRanges", DataIn, DataOu
// we have selection which contains two blocks:
int firstSelChar1 = DataOut[0];
int countSelChars1 = DataOut[1];
int firstSelChar2 = DataOut[2];
int countSelChars2 = DataOut[3];
...
```

For simplify in/out operations with **SAFEARRAY** you can use the IPDFXCargs:

```
...
// obtain a new IPDFXCargs object with array of numbers:
DoDocumentVerb(docId, "Pages[0].Text", "GetSelectedRanges", DataIn,
               DataOut, PXCVA_OutArgs);
IPDFXCargsObj args = DataOut;
int firstSelChar1 = args[0];
int countSelChars1 = args[1];
int firstSelChar2 = args[2];
int countSelChars2 = args[3];
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text

Highlights all characters from the specified range on the page.

Name

```
Highlight
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The index of the first character to highlight.
2	Yes	LONG	The count of characters to highlight.
3	No	LONG	The color (COLORREF) of the highlighted comment. By default means **#00FFFF** (yellow).
4	No	Double	The opacity of the highlight comment. By default set to **1.0** (opaque).
5	No	BSTR	The name of the new annotation object (may be used for identification of annotation on the page in the future)

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// highlight first 10 characters on first page, violet, 65% opacity:
DoVerb("Documents[0].Pages[0].Text", "Highlight",
            DataIn(0, 10, #800080, 0.65), DataOut, 0);
// highlight all characters on first page:
DoVerb("Documents[0].Pages[0].Text", "Highlight", DataIn(0, -1), DataOut,
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text

Selects all characters from a specified range on the page.

Name

```
Select
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The index of the first character to select.
2	Yes	LONG	The count of characters to select.
3	No	LONG	The flag to ensure visibility of this selection. By default set to **0**.

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// select first 10 characters on first page, and make visible new selectic
DoVerb("Documents[0].Pages[0].Text", "Select", DataIn(0, 10, 1), DataOut,
```

```
// select all characters on first page:
DoVerb("Documents[0].Pages[0].Text", "Select", DataIn(0, -1), DataOut, 0);
// deselect all characters on first page:
DoVerb("Documents[0].Pages[0].Text", "Select", DataIn(0, 0), DataOut, 0);
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text

Strikes out all characters from specified range on the page.

Name

```
StrikeOut
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The index of the first character to strike out.
2	Yes	LONG	The count of characters to strike out.
3	No	LONG	The color (COLORREF) of the 'strike out' comment. By default is set to **#0000FF** (red).
4	No	Double	The opacity of the 'strike-out' comment. By default is set to **1.0** (opaque).
5	No	BSTR	The name of the new annotation object (may be used for identification of annotation on the page in the future)

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// strike out first 10 characters on first page, violet, 65% opacity
DoVerb("Documents[0].Pages[0].Text", "StrikeOut",
            DataIn(0, 10, #800080, 0.65), DataOut, 0);
// strike out all characters on first page
DoVerb("Documents[0].Pages[0].Text", "StrikeOut", DataIn(0, -1), DataOut,
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text

Underlines all characters from the specified range on the page.

Name

```
Underline
```

Arguments

#	Req.	Type	Description
1	Yes	LONG	The index of the first character to underline.
2	Yes	LONG	The count of characters to underline.
3	No	LONG	The color (COLORREF) of the underlined comment. By default means **#00FF00** (green).
4	No	Double	The opacity of the underlined comment. By default is set to **1.0** (opaque).
5	No	LONG	The squiggly underline type. By default means **0**.
6	No	BSTR	The name of the new annotation object (may be used for identification of annotation on the page in the future)

Return Value

Returns **S_OK** on success, or an error value otherwise. To obtain the text description of a received error code, use IPDFXCview::GetTextFromResult.

Remarks

For example (in pseudocode):

```
...
// underline first 10 characters on first page, violet, 65% opacity, squig
DoVerb("Documents[0].Pages[0].Text", "Underline",
            DataIn(0, 10, #800080, 0.65, 1), DataOut, 0);
// underline all characters on first page:
DoVerb("Documents[0].Pages[0].Text", "Underline", DataIn(0, -1), DataOut,
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text

This object represents an array of all characters on the page.

Item Template

<Item> - defines item template for each character.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains the total number of all characters on the page.

Remarks

For example (in pseudocode):

```
...
// gets chars count on the first page:
DoVerb("Documents[0].Pages[0].Text.Chars.Count", "get", DataIn, DataOut, 0
// gets first char on the first page:
DoVerb("Documents[0].Pages[0].Text.Chars[0].Code", "get", DataIn, DataOut,
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text

Defines an item template for each character object.

Contained Objects

Name	Type	Description
Code	Integer [R]	Represents Unicode code.
Style	Group [R]	Represents character style.
Quad	Group [R]	Contains character coordinates on page.

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text.Chars

This object represents the character style.

Contained Objects

Name	Type	Description
Font	Group [R]	Represents information about the font of this character.
FontSize	Double [R]	Font size.
FColor	Integer [R]	Fill color.
SColor	Integer [R]	Strike color.
HScale	Double [R]	Horizontal scaling.
RMode	Integer [R]	Rendering mode.
Rise	Double [R]	Base line offset (text rise).

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text.Chars::<Item>

This object represents the Font object for the specified character.

Contained Objects

Name	Type	Description
ID	Integer [R]	Font's unique ID.
Name	String [R]	Font name.

Ascent	Double [R]	Font ascent.
Descent	Double [R]	Font descent.
Embedded	Integer/ String [R]	The font embedded flag. For possible values, see the Booleans.
BBox	Group [R]	Represents the bound box of the font. Has the Objects::RectangleF structure.

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text.Chars::<Item>.Style

This object represents the array for all words on the page.

Item Template

<Item> - defines item template for each word.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains the number count of all words on the page.

Remarks

For example (in pseudocode):

```
...
// gets words count on the first page:
DoVerb("Documents[0].Pages[0].Text.Words.Count", "get", DataIn, DataOut, 0
// gets first word on the first page:
DoVerb("Documents[0].Pages[0].Text.Words[0].String", "get", DataIn, DataOu
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text

Defines an item template for each word.

Contained Objects

Name	Type	Description
String	String [R]	Represents word text.
Offset	Integer [R]	Represents the word offset position on the page (index of first character).
Length	Integer [R]	Represents the word length (character count).
Quads	Group [R]	Contains word coordinates on page.

See Also

Objects::Documents,
Objects::Documents::<Item>.Pages::<Item>.Text.Words

This object represents the array for all text lines on the page.

Item Template

<Item> - defines item template for each line.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains the number count of all text lines on the page.

Remarks

For example (in pseudocode):

```
...
// gets lines count on the first page:
DoDocumentVerb(docId, "Pages[0].Text.Lines.Count", "get", DataIn, DataOut,
// gets first line offset/length:
DoDocumentVerb(docId, "Pages[0].Text.Lines[0].Offset", "get", DataIn, Data
offs = DataOut;
DoDocumentVerb(docId, "Pages[0].Text.Lines[0].Length", "get", DataIn, Data
len = DataOut;
// gets text of first line:
DoDocumentVerb(docId, "Pages[0].Text", "get", DataIn(offs, len), DataOut,
str = DataOut;
...
// gets quad(s) of line:
DoDocumentVerb(docId, "Pages[0].Text.Lines[0].Quads.Value", "get", DataIn,
// bottom/left point of first quad:
q[0].x = DataOut[0];
q[0].y = DataOut[1];
// bottom/right point of first quad:
q[1].x = DataOut[2];
q[1].y = DataOut[3];
// top/right point of first quad:
q[2].x = DataOut[4];
q[2].y = DataOut[5];
// top/left point of first quad:
q[3].x = DataOut[6];
q[3].y = DataOut[7];
...
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text

Defines an item template for each text line.

Contained Objects

Name	Type	Description
Type	String [R]	Represents type of text line. Possible values: **0** - simple text line, all characters are on one line. **1** - composite text line, all characters are on one

		curve.
Offset	Integer [R]	Represents the line offset position on the page (index of first character).
Length	Integer [R]	Represents the line length (character count).
Quads	Group [R]	Contains line coordinates on page.

See Also

Objects::Documents,
Objects::Documents::<Item>.Pages::<Item>.Text.Lines

2.2.2.4.1.3 Form

Allows to specify settings for PDF form for the document.

Contained Objects

Name	Type	Description
HighlightFields	Group [RW]	Defines form fields highlight settings.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

Allows to specify form fields highlight settings for the document.

Contained Objects

Name	Type	Description
Visible	Integer/ String [RW]	Specifies visibility of the form highlight. For possible named values, see the Booleans. Default value is **0**.
Mask	Integer/ String [RW]	Specifies initial highlight mask to highlight selected form fields. For possible named values, see Highlight Form Fields Masks. This value can be a combination of the mask integer values.
RequiredOnly	Integer/ String [RW]	Highlight only required form fields. For possible named values, see the Booleans. Default value is **0**.

Remarks

To set color of the highlight see Objects::Forms.HighlightFields.

Examples for usage (in pseudocode):

1. Set highlight color:

```
SetProperty("Forms.HighlightFields.FillColor", "Blue", 0);
SetProperty("Forms.HighlightFields.BorderColor", "Red", 0);
SetProperty("Documents[#4095].Form.HighlightFields.Visible", "true", 0);
```

2.1 Highlight only CheckBoxes fields:

```
SetProperty("Documents[#4095].Form.HighlightFields.Mask", "CheckBoxes", 0)
SetProperty("Documents[#4095].Form.HighlightFields.Visible", "true", 0);
```

2.2 Highlight CheckBoxes and TextBoxes fields:

```
SetProperty("Documents[#4095].Form.HighlightFields.Mask", 0x00000008 | 0x0
SetProperty("Documents[#4095].Form.HighlightFields.Visible", "true", 0);
```

See Also

IPDFXCview::GetProperty, IPDFXCview::SetProperty,
Objects::Documents::<Item>.Form
Objects::Forms.HighlightFields

2.2.2.4.1.4 View

This is the common object that contains the document view attributes: visibility, positions, layouts for document bars, panes, windows. Also allows customizing of the document toolbars.

Contained Objects

Name	Type	Description
Bars	Array [RWS]	Represents all UI bars (menu, command, status bars). For details about document bars, see the Document Bars.
Panes	Array [RWS]	Represents all UI panes (named containers for special views). For details about document panes, see the Document Panes.
ShowScrollBars	Integer/ String [RW]	Controls visibility of scroll bars in pages view. For possible named values, see the Booleans. Default value is **1**.

Remarks

Examples for usage (in pseudocode). *4095* is an example target document's unique ID.

1.1. get bars count (same as panes):

```
...
DoVerb("Documents[#4095].View.Bars.Count", "get", DataIn, DataOut, 0);
// or:
GetProperty("Documents[#4095].View.Bars.Count", DataOut, 0);
// or:
GetDocumentProperty(4095, "View.Bars.Count", DataOut, 0);
...
```

1.2. get bar unique name by order index (same as panes):

```
...
DoVerb("Documents[#4095].View.Bar[0].Name", "get", DataIn, DataOut, 0);
// or:
GetProperty("Documents[#4095].View.Bar[0].Name", DataOut, 0);
// or:
GetDocumentProperty(4095, "View.Bar[0].Name", DataOut, 0);
...
```

1.3. get 'options' bar visibility:

```
...
DoVerb("Documents[#4095].View.Bars["Options"].Visible",
                "get", DataIn, DataOut, 0);
// or:
```

```
GetProperty("Documents[#4095].View.Bars["Options"].Visible",
            DataOut, 0);
// or:
GetDocumentProperty(4095, "View.Bars["Options"].Visible",
            DataOut, 0);
...
```

1.4. get thumbnails pane visibility:

```
...
DoVerb("Documents[#4095].View.Panes["Thumbnails"].Visible", "get",
            DataIn, DataOut, 0);
// or:
GetProperty("Documents[#4095].View.Panes["Thumbnails"].Visible",
            DataOut, 0);
// or:
GetDocumentProperty(4095, "View.Panes["Thumbnails"].Visible",
            DataOut, 0);
...
```

2.1. show 'options' toolbar:

```
...
DoVerb("Documents[#4095].View.Bars["Options"].Visible",
            "set", DataIn(1), DataOut, 0);
// or:
SetProperty("Documents[#4095].View.Bars["Options"].Visible",
            DataIn(1), 0);
// or:
SetDocumentProperty(4095, "View.Bars["Options"].Visible",
            DataIn(1), 0);
...
```

2.2. show thumbnails pane:

```
...
DoVerb("Documents[#4095].View.Panes["Thumbnails"].Visible", "set",
            DataIn(1), DataOut, 0);
// or:
SetProperty("Documents[#4095].View.Panes["Thumbnails"].Visible",
            DataIn(1), 0);
// or:
SetDocumentProperty(4095, "View.Panes["Thumbnails"].Visible",
            DataIn(1), 0);
...
```

See Also

Object Name Notation,
Objects::Documents,
Objects::View

2.2.2.5 Export

Contains all supported properties for exporting documents to an external image and text formats. See also IPDFXCview::ExportDocument.

Contained Objects

Name	Type	Description
Mode	Integer/ String [RWS]	Defines the document export mode. For possible values, see the Export Modes. Default value is **0** (export document pages to images).
Image	Group [RWS]	Defines document exporting to image(s) options.

See Also

IPDFXCview::ExportDocument, IPDFXCview::DoVerb,
IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.5.1 Image

Defines options for exporting document pages to images.

Contained Objects

Name	Type	Description
Type	Integer/ String [RWS]	Specifies the exported image type. For possible values, see the Image Types. Default value is **0x424D5020** (BMP).
Mode	Integer/ String [RWS]	Specifies how to combine pages into an image file. For possible values, see the Export to Image Modes . Default value is **0** (all pages into one multipage image file).
RangeType	Integer/ String [RWS]	Defines the type of pages range. For possible values, see the Range Types. Default value is **1** (all pages).
RangeFilter	Integer/ String [RWS]	Specifies additional filter for the pages range. For possible values, see the Range Filters. Default value is **1** (all pages).
RangeReverse	Integer/ String [RWS]	Specifies whether to export selected pages in reverse order. For possible named values, see the Booleans. Default value is **0**.
RangeText	String [RWS]	Specifies the exact pages to export. The string should contain the page numbers and/or page ranges separated by commas counting from the start of the document. For example: **"1, 3, 5-12"**.
Resolution	Integer [RWS]	Specifies the resolution of the exported pages in dots per inch. Default value is **300** (dpi).
Background	Integer/ String [RWS]	Specifies the background color for the exported pages as a **COLORREF** value. If highest byte of this value is equal to **0** then the background is transparent. For possible named values, see the Colors. Default value is **0xFFFFFFFF** (white).
FolderName	String [RWS]	Specifies image export folder name. When multiple image files are produced by the export operation, this folder receives all exported image files, and image file names are automatically generated. If this value is empty then the <My Documents> standard folder will be used automatically. Default value is empty (NULL or "").
FileName	String [RWS]	Specifies image export file name. It supports macros for generic names. See Name Generation Macros.

See Also

Image Types,
Export to Image Modes,
IPDFXCview::ExportDocument

2.2.2.6 Find

Contains all supported properties for simple text searching.

Contained Objects

Name	Type	Description
Text	String [RWS]	Specifies text for searching in active PDF document.
Options	Group [RWS]	Specifies find options.

Remarks

To start find text in active document you may execute **"FindNext"** UI-command.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Commands List

2.2.2.6.1 Options

Contains options for simple text search.

Contained Objects

Name	Type	Description
WholeWordsOnly	Integer/String [RWS]	Search only whole words. For possible named values, see the Booleans. Default value is **0**.
CaseSensitive	Integer/String [RWS]	Use case sensitive search. For possible named values, see the Booleans. Default value is **0**.
IncludeBookmarks	Integer/String [RWS]	Search in bookmarks. For possible named values, see the Booleans. Default value is **0**.
IncludeComments	Integer/String [RWS]	Search in comments. For possible named values, see the Booleans. Default value is **0**.
DenyNotFoundMessage	Integer/String [RWS]	Hide message which informs about end of search. For possible named values, see the Booleans. Default value is **0**.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.7 Forms

Allows to specify settings for PDF forms.

Contained Objects

Name	Type	Description
HighlightFields	Group [RWS]	Defines initial form fields highlight settings.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.7.1 HighlightFields

Allows to specify initial settings for form highlighting.

Contained Objects

Name	Type	Description
UseInitial	Integer/ String [RWS]	When specified, form highlight settings for every new opened document will be taken from initial highlight settings. For possible named values, see the Booleans. Default value is **0**.
InitialMask	Integer/ String [RWS]	Specifies initial highlight mask to highlight selected form fields. For possible named values, see Highlight Form Fields Masks.
InitialRequiredOnly	Integer/ String [RWS]	Highlight only required form fields. For possible named values, see the Booleans. Default value is **0**.
FillColor	Integer/ String [RWS]	Specifies fill color for the form highlight. For possible named values, see the Colors. Default value is **RGB(204,215,255)**.
BorderColor	Integer/ String [RWS]	Specifies border color for the form highlight. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).

Remarks

To set form highlight settings for specified document, see Objects::Documents::<Item>.Form.HighlightFields.

See Also

IPDFXCview::GetProperty, IPDFXCview::SetProperty, Objects::Documents::<Item>.Form.HighlightFields, Objects::Forms

2.2.2.8 General

Accesses the main properties of the ActiveX control.

Contained Objects

Name	Type	Description
AllowAllAccelerators	Integer/ String [RWS]	Enables/disables keyboard accelerators. For possible named values, see the Booleans. Default value is **0** (all accelerators denied).

ApplicationModulePath	String [RW]	Contains the full file name of exe-module (PDFXCview.exe) of our ActiveX control.
ApplicationTitle	String [RW]	Contains the application title. This string will be displayed in more titles, dialogs, messages etc. Default value is **"PDF-XChange Viewer Control"**.
DenyAllContextMenus	Integer/ String [RWS]	Allow/deny displaying of context menus displaying. For possible named values, see the Booleans. Default value is **0** (all context menus allowed).
DenyAllExportOperations	Integer/ String [RWS]	Enables/disables export ability. For possible named values, see the Booleans. Default value is **0** (all export operations allowed).
DenyAllModifyOperations	Integer/ String [RWS]	Allow/deny all modify-operations for all documents. For possible named values, see the Booleans. Default value is **0** (modifications allowed).
DenyAllPrintOperations	Integer/ String [RWS]	Allow/deny all print-operations for all documents. For possible named values, see the Booleans. Default value is **0** (printing allowed).
DenyAllSaveOperations	Integer/ String [RWS]	Allow/deny all save-operations for all documents. For possible named values, see the Booleans. Default value is **0** (saving allowed).
DenyOpenDocumentsWhenDrop	Integer/ String [RWS]	Allow/deny open PDF document with drag and drop. For possible named values, see the Booleans. Default value is **0** (open document with Drag&Drop allowed).
DenyProgressDisplaying	Integer/ String [RWS]	Allow/deny progress dialog displaying. For possible named values, see the Booleans. Default value is **0** (progress displaying allowed).
DocumentInterface	Integer/ String [RWS]	Defines the UI interface of documents. For possible values, see the Document Interfaces. Default value is **1** (Single Document Interface).
ProcessID	Integer/ String [R]	Retrieves the server process identifier. (process ID of PDFXCview.exe module).
ProcessName	String [R]	Retrieves the server process name. (name of PDFXCview.exe module).

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.9 International

Controls UI localization.

Contained Objects

Name	Type	Description
LocaleID	Integer/ String [RWS]	Defines the standard locale identifier (LCID). The specified language will be used in all UI elements (in menu, toolbars, panes, dialogs, messages, etc.).

		For possible named values, see the User Interface Languages. Default value is -**1**("Default") (selects the *default* control's language). For select the control's *built-in* language exactly you should specify the **-2**("BuiltIn") value.
DefaultLocaleID	Integer/ String [RWS]	By default the control's *built-in* language is *default*. This property specifies(overrides) the control's default language. If it is equal to any of supported locale identifiers then it sets a new default language which differs from control's built-in language. For possible named values, see the User Interface Languages. Default value is **-2**("BuiltIn"). In this case the control selects the built-in language (English(US)) when **LocaleID** is equal to **-1** ("Default").

Remarks

This possibility can be used for supported languages only. To find all currently-supported languages examine the contents of **"<FolderOfExeModule>\Languages\"** folder. See also Objects::General.ApplicationModulePath.

Example for usage (in pseudocode):

1. get current UI language ID:

```
...
GetProperty("International.LocaleID", DataOut, 0);
// result:
DataOut == 0x0409; // i.e: current language is 'English(US)'
...
```

2. set new UI language:

```
...
SetProperty("International.LocaleID", DataIn("French(FR)"), 0);
// or:
SetProperty("International.LocaleID", DataIn(0x040C), 0);
...
```

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.10 Notifications

Allows to receive and modify control's events.

Contained Objects

Name	Type	Description
Mouse	Group [RWS]	Allows to get information about mouse events.
ContextMenu	Group [RWS]	Used for confirmation and customization of context menu before it displaying.
Selection	Group [RWS]	Informs about any selection changes in the document.
Print	Group [RWS]	Informs about printing of the document.

TextEditor	Group [RWS]	Informs about text editing in the document.
Keyboard	Group [RWS]	Allows to get information about keyboard events.
FieldChanged	Group [R]	Informs you about changing value of form field.
BeforeSaveDoc	Group [RW]	Control sends this event before saving of document.
DocSaved	Group [R]	Control sends this event after saving of document.
BeforeCloseDoc	Group [RW]	Control sends this event before closing of document.
DocClosed	Group [R]	Control sends this event when document is closed.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.10.1 BeforeCloseDoc

Control sends this event before closing of document.

Contained Objects

Name	Type	Description
DocID	Integer [R]	Retrieves unique ID of the document.
CancelAllowed	Integer/ String [R]	Means **1** if canceling of closing document is allowed. For possible named values, see the Booleans.
Cancel	Integer [RW]	Allows to stop closing of the document, depends of **CancelAllowed**. For possible named values, see the Booleans. Default value is **0** (false).

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, _IPDFXCviewEvents::OnEvent

2.2.2.10.2 BeforeSaveDoc

Control sends this event before saving of document.

Contained Objects

Name	Type	Description
DocID	Integer [R]	Retrieves unique ID of the document.
SaveMode	Integer/ String [R]	Retrieves document save mode identifier. See Document Save Modes.
DestFileName	String [R/W]	Retrieves destination file name and you may change it to other valid file name. If document is being saved in a stream then this value will be empty.

Cancel	Integer [R]	Allows to stop saving of the document. For possible named values, see the Booleans. Default value is **0** (false).

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, _IPDFXCviewEvents::OnEvent

2.2.2.10.3 ContextMenu

Used for confirmation and customization of context menu before it displaying.

Contained Objects

Name	Type	Description
x	Integer [R]	Retrieves screen x position of the popup menu.
y	Integer [R]	Retrieves screen y position of the popup menu.
MenuName	String [R]	Retrieves name of context menu.
UserChoice	Integer/ String [RW]	Allows to deny showing default context menu. For possible named values, see the Context Menu User Choices. This parameter can be also set to specific command ID (see also Objects::Commands) which will run instead of context menu. Default value is **0** (show default context menu).

Remarks

Example for usage(in pseudocode), add Sticky Note annotation instead of Hand context menu:

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
   if (Type == PXCVA_OnNamedNotify) and
      (Name == "Notifications.ContextMenu") then
   {
      GetProperty("Notifications.ContextMenu.MenuName", DataOut, 0);
      if (DataOut == "Hand") then
      {
         SetProperty("Notifications.ContextMenu.UserChoice", 33135, 0);
      }
   }
}
...
```

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, _IPDFXCviewEvents::OnEvent

2.2.2.10.4 DocClosed

Control sends this event after close of document.

Contained Objects

Name	Type	Description
DocID	Integer [R]	Retrieves unique ID of the document which has been closed.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, _IPDFXCviewEvents::OnEvent

2.2.2.10.5 DocSaved

Control sends this event after saving of document.

Contained Objects

Name	Type	Description
DocID	Integer [R]	Retrieves unique ID of the document.
SaveMode	Integer/ String [R]	Retrieves document save mode identifier. See Document Save Modes.
DestFileName	String [R]	Retrieves destination file name. If document is saved in a stream then this value will be empty.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, _IPDFXCviewEvents::OnEvent

2.2.2.10.6 FieldChanged

Informs you about changing value of form field.

Contained Objects

Name	Type	Description
DocID	Integer [R]	Retrieves unique ID of document.
Name	String [R]	Retrieves name of the changed form field.
PageIndex	Integer [R]	Retrieves index of page with the form filed (corresponding widget - visual object of the form field).
IndexOnPage	Integer [R]	Retrieves index of widget on the page.

Remarks

Example for usage(in pseudocode):

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
   if (Type == PXCVA_OnNamedNotify) and
      (Name == "Notifications.FieldChanged") then
   {
      GetProperty("Notifications.FieldChanged.Name", DataOut, 0);
      if (DataOut == "Text1.0") then
```

```
            {
               // get actual value, by special java script:
               string js = "var f = this.getField(" + DataOut + "); if (f != nul
               string fieldValue; // clear
               RunJavaScript(js, out fieldValue, ...);
               ...
            }
         }
      }
      ...
```

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, _IPDFXCviewEvents::OnEvent

2.2.2.10.7 Keyboard

Allows to get information about keyboard events.

Contained Objects

Name	Type	Description
Filter	Integer/ String [RWS]	Specifies filter for receiving keyboard events. For possible named values, see the Keyboard Notifications Filter Flags. You may use a combination of one or more of the following values. Default value is **0** (None)
Msg	Integer [R]	Specifies Widows keyboard messages: WM_KEYDOWN, WM_SYSKEYDOWN, WM_CHAR ...
Code	Integer [R]	Specifies the virtual key code of the given key.
RepCnt	Integer [R]	Repeat count (the number of times the keystroke is repeated as a result of the user holding down the key).
Flags	Integer [R]	Specifies the scan code, key-transition code, previous key state, and context code.
Skip	Integer/ String [RW]	Allows to deny processing current keyboard message by Viewer. For possible named values, see the Booleans. Default value is **0** (false).

Remarks

Example for usage (in pseudocode):

1. enable receiving all keyboard events:

```
...
SetProperty("Notifications.Keyboard.Filter", "All",0);
...
```

2. ignore Enter pressed event:

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
   if (Type == PXCVA_OnNamedNotify) and
      (Name == "Notifications.Keyboard") then
   {
```

```
            // Get keyboard event message
            GetProperty("Notifications.Keyboard.Code", vDataOut, 0);
            if (vDataOut == 13) then
            {
               // Ignore this message
               SetProperty("Notifications.Keyboard.Skip", "true", 0);
            }
         }
      }
      ...
```

2.2.2.10.8 Mouse

Allows to get information about mouse events.

Contained Objects

Name	Type	Description
Filter	Integer/ String [RWS]	Specifies filter for receiving mouse events. For possible named values, see the Mouse Notifications Filter Flags. You may use a combination of one or more of the following values. Default value is **0** (None)
x	Integer [R]	Retrieves x position of the cursor in screen coordinates.
y	Integer [R]	Retrieves y position of the cursor in screen coordinates.
Msg	Integer [R]	Retrieves mouse event message.
TargetName	String [R]	Retrieves UI component name under mouse cursor.
Skip	Integer/ String [RW]	Allows to deny processing current mouse message by Viewer. For possible named values, see the Booleans. Default value is **0** (false).
WheelDelta	Integer [R]	If mouse wheel was rotated then this parameter indicates the distance the wheel was rotated. A positive value indicates that the wheel was rotated forward, away from user. A negative value indicates that the wheel was rotated backward, toward the user.
Document	Integer [R]	Retrieves document ID under mouse cursor.
Page	Integer [R]	Retrieves page number under mouse cursor.

Remarks

Example for usage (in pseudocode):

1. enable receiving all mouse events:

```
...
SetProperty("Notifications.Mouse.Filter", "All",0);
...
```

2. ignore left mouse button down event:

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
```

```
{
   if (Type == PXCVA_OnNamedNotify) and
      (Name == "Notifications.Mouse") then
   {
      // Get mouse event message
      GetProperty("Notifications.Mouse.msg", vDataOut, 0);
      if (vDataOut == WM_LBUTTONDOWN) then
      {
         // Ignore this message
         SetProperty("Notifications.Mouse.Skip", "true", 0);
      }
   }
}
...
```

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, _IPDFXCviewEvents::OnEvent, Objects::Documents::<Item>.Pages::<Item>.TranslateScreenPoint

2.2.2.10.9 Print

Informs about printing of the document.

Contained Objects

Name	Type	Description
Filter	Integer/ String [RWS]	Specifies filter for receiving selection changes events. For possible named values, see the Print Notifications Filter Flags. You may use a combination of one or more of the following values. Default value is **0** (None)
Type	Integer/ String [R]	Represents type of the print-notification. Contains one value from the Print Notifications Filter Flags enumeration (except "None" and "All" values).
PrinterName	String [R]	Public name of the printer which is being used for printing of the document.
Document	Integer [R]	Contains the ID of the document which is being printed.
Sheet	Integer [R]	Contains the index(0-based) of the sheet which is printed now. Can be undefined(-1) for some print-notifications.
SheetsCount	Integer [R]	Contains the exact quantity of sheets which will be printed out.
Page	Integer [R]	Contains the index(0-based) of the document's page which is printed now. Can be undefined(-1) for some print-notifications.
PagesCount	Integer [R]	Contains the exact quantity of the document's pages which will be printed out.
Copies	Integer [RW]	Number of copies. *Can be changed at Type="Prepare" stage only.*
Collate	Integer/ String [RW]	Flag to collate copies. *Can be changed at Type="Prepare" stage only.* For possible named values, see the Booleans. Default value is **0** (false).
Cancel	Integer/	Allows to break printing process.

	String [RW]	For possible named values, see the Booleans. Default value is **0** (false).

Remarks

Example for usage(in pseudocode):

1. enable receiving all printing events:

```
...
SetProperty("Notifications.Print.Filter", "All",0);
...
```

2. catch selection changes and check selection state for the document:

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
   if (Type == PXCVA_OnNamedNotify) and
      (Name == "Notifications.Print") then
   {
      // obtain notifications type(print stage):
      GetProperty("Notifications.Print.Type", DataOut, PXCVA_GetNamed);
      if (DataOut == "EndDocument")
      {
         // obtain printed document:
         GetProperty("Notifications.Print.Document", DataOut, 0);
         ...
      }
   }
}
...
```

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, _IPDFXCviewEvents::OnEvent

2.2.2.10.10 Selection

Informs about any selection in the document.

Contained Objects

Name	Type	Description
Filter	Integer/ String [RWS]	Specifies filter for receiving selection changes events. For possible named values, see the Selection Notifications Filter Flags. You may use a combination of one or more of the following values. Default value is **0** (None)
Document	Integer [R]	Contains the ID of the document in which selection was changed.

Remarks

Example for usage(in pseudocode):

1. enable receiving all selection events:

```
...
SetProperty("Notifications.Selection.Filter", "All",0);
...
```

2. catch selection changes and check selection state for the document:

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
   if (Type == PXCVA_OnNamedNotify) and
      (Name == "Notifications.Selection") then
   {
      GetProperty("Notifications.Selection.Document", DataOut, 0);
      int docID = DataOut;
      DoDocumentVerb(docID, NULL, "GetSelectionState", DataIn, DataOut, 0)
      if (DataOut != 0)
      {
        // we have selection in this document
      }
      ...
   }
}
...
```

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, _IPDFXCviewEvents::OnEvent

2.2.2.10.11 TextEditor

Informs about text editing in the document.

Contained Objects

Name	Type	Description
Filter	Integer/ String [RWS]	Specifies filter for receiving text editing events. For possible named values, see the Text Editor Notifications Filter Flags. You may use a combination of one or more of the following values. Default value is **0** (None)
Type	Integer/ String [R]	Represents type of the text-editor notification. Contains one value from the Text Editor Notifications Filter Flagsenumeration (except "None" and "All" values).
Document	Integer [R]	Contains the ID of the document with the active text-editor module.

Remarks

Example for usage(in pseudocode):

1. enable receiving all text-editor events:

```
...
SetProperty("Notifications.TextEditor.Filter", "All",0);
...
```

2. catch selection changes and check selection state for the document:

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
   if (Type == PXCVA_OnNamedNotify) and
      (Name == "Notifications.TextEditor") then
   {
      // obtain notifications type:
      GetProperty("Notifications.TextEditor.Type", DataOut, PXCVA_GetNamed
      if (DataOut == "End")
```

```
            {
               // obtain changed document:
               GetProperty("Notifications.TextEditor.Document", DataOut, 0);
               ...
            }
         }
      }
      ...
```

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, _IPDFXCviewEvents::OnEvent

2.2.2.11 Operations

Allows to specify settings for operations.

Contained Objects

Name	Type	Description
DeletePages	Group [RWS]	Allows to specify settings for pages deletion from PDF document.
ExtractPages	Group [RWS]	Allows to extract selected pages from the current PDF to one or more PDF files.
InsertEmptyPages	Group [RWS]	Allows to specify settings for insertion of empty pages to PDF document.
InsertPages	Group [RWS]	Allows to specify settings for inserting pages from the source document into the current document.
NewDocument	Group [RWS]	Allows to create new PDF Document.
RotatePages	Group [RWS]	Allows to specify settings for page rotation.
SummarizeAnnots	Group [RWS]	Allows to get a summary all the comments associated with a PDF to view them as a new PDF Document, Rich Text Format, Plain Text or Web Page.

See Also

IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.11.1 DeletePages

Allows to specify settings for pages deletion from PDF document.

Contained Objects

Name	Type	Description
RangeType	Integer/ String [RW]	Defines the type of page range for page deletion. For possible values, see the Range Types. Supports only **Selected**, **Current** and **Exact** values. Default value is **1** (all pages).
RangeFilter	Integer/ String [RW]	Specifies an additional filter for the page range. For possible values, see the Range Filters. Default value is **1** (all pages).
RangeText	String	Specifies the exact pages for deletion. The string

	[RW]	should contain the page numbers and/or page ranges separated by commas counting from the start of the document. For example: **"1, 3, 5-12"**.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Operations::DeleteDocumentPages, Objects::Documents::<Item>::DeletePages

2.2.2.11.2 ExtractPages

Allows to specify settings for extracting pages from PDF document.

Contained Objects

Name	Type	Description
RangeType	Integer/ String [RW]	Defines the type of page range for extraction. For possible values, see the Range Types. Supports only **All**, **Selected**, **Current** and **Exact** values. Default value is **1** (all pages).
RangeFilter	Integer/ String [RW]	Specifies an additional filter for the page range. For possible values, see the Range Filters. Default value is **1** (all pages).
RangeText	String [RW]	Specifies the exact pages for extraction. The string should contain the page numbers and/or page ranges separated by commas counting from the start of the document. For example: **"1, 3, 5-12"**.
ExtractToFiles	Integer/ String [RWS]	Extract pages into the new external file(s). For possible named values, see the Booleans. Default value is **0** (false).
ToOneFile	Integer/ String [RWS]	If means **"true"** then specifies to extract selected pages into the one external file, otherwise each page will be extracted into the separated file. This option can be used if **ExtractToFiles** is enabled**.** For possible named values, see the Booleans. Default value is **0** (each page to the new file).
DestFolder	String [RWS]	Specifies the fully-qualified name of the folder to receive the new single-page PDF files.
DestFileName	String [RWS]	Specifies a macro-based file name to use for the created files. It supports macros for generic names. See Name Generation Macros.
DeleteAfterExtract	Integer/ String [RWS]	When enabled, the pages will be extracted to separate files named and located according to **DestFileName** and **DestFolder** properties. For possible named values, see the Booleans. Default value is **0** (false).

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Operations::ExtractDocumentPages, Objects::Documents::<Item>::ExtractPages

2.2.2.11.3 InsertEmptyPages

Allows to specify settings for insertion of empty pages to PDF document.

Contained Objects

Name	Type	Description
PaperWidth	Double [RWS]	Specifies new pages width in points. Default value is **595 pt** (210 mm).
PaperHeight	Double [RWS]	Specifies new pages height in points. Default value is **842 pt** (297 mm).
PaperMode	Integer/ String [RWS]	Specifies what paper size to use. Can use custom sizes from **PaperWidth** and **PaperHeight** parameters or standard paper type specified by **PaperName**. For possible named values, see the Paper Modes. Default value is **1** (standard).
PaperName	String [RWS]	The name of the standard type of paper. Default value is **A4**.
Orientation	Integer/ String [RWS]	Specifies landscape paper orientation for new document. For possible named values, see the Booleans. Default value is **0**.
Count	Integer [RWS]	Specifies pages count for new document. Default value is **1**.
InsertBefore	Integer [RWS]	Specifies the zero-based index of the page to insert the empty pages before.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Operations::InsertEmptyDocumentPages, Objects::Documents::<Item>::InsertEmptyPages

2.2.2.11.4 InsertPages

Allows to specify settings for inserting pages from the source document into the current document.

Contained Objects

Name	Type	Description
RangeType	Integer/ String [RW]	Defines the type of page range for insertion. For possible values, see the Range Types. Supports only **All** and **Exact** values. Default value is **1** (all pages).
RangeFilter	Integer/ String [RW]	Specifies an additional filter for the page range. For possible values, see the Range Filters. Default value is **1** (all pages).
RangeText	String [RW]	Specifies the exact pages of source document for insertion. The string should contain the page numbers and/or page ranges separated by commas counting from the start of the document. For example: **"1, 3, 5-12"**.
FromExternal	Integer/	If set to true then external document will be used as

	String [RW]	source. Path to this document is specified in **File** property. If set to false then pages will be inserted from already opened document. ID of source document is specified in **DocID** property. For possible named values, see the Booleans. Default value is **1** (true).
File	String [RW]	Specifies the path to source document. This path will be taken only if **FromExternal** is set to **true**.
DocID	String [RW]	Specifies already opened document ID from which pages will be inserted. This will work only when **FromExternal** is set to **false**. This parameter cannot be equal to current document ID.
InsertBefore	Integer [RWS]	Specifies zero-based index of the page to insert the source document pages before.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Operations::InsertDocumentPages, Objects::Documents::<Item>::InsertPages

2.2.2.11.5 NewDocument

Allows to specify settings for creation of new PDF document.

Contained Objects

Name	Type	Description
FromBlank	Group [RWS]	Specifies options for creating new empty document.
SpecVersion	Integer/ String [RWS]	Specifies version of **PDF-Specification** for creating new document. For possible named values, see the PDF-Specification Versions. Default value is **0**(Auto).

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.11.5.1 FromBlank

Allows to set settings for creating new empty PDF document.

Contained Objects

Name	Type	Description
PaperWidth	Double [RWS]	Specifies new PDF page width in points. Default value is **595 pt** (210 mm).
PaperHeight	Double [RWS]	Specifies new PDF page height in points. Default value is **842 pt** (297 mm).
PaperMode	Integer/ String [RWS]	Specifies what paper size to use. Can use custom sizes from **PaperWidth** and **PaperHeight** parameters or standard paper type

		specified by **PaperName**. For possible named values, see the Paper Modes. Default value is **1** (standard).
PaperName	String [RWS]	The name of the standard type of paper. Default value is **A4**.
Orientation	Integer/ String [RWS]	Specifies landscape paper orientation for new document. For possible named values, see the Booleans. Default value is **0**.
Count	Integer [RWS]	Specifies pages count for new document. Default value is **1**.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Operations::CreateNewBlankDocument

2.2.2.11.5.2 FromImages

Allows to set settings for creating new PDF document from images.

Contained Objects

Name	Type	Description
Paper	Group [RWS]	Specifies new PDF-page options.
Layout	Group [RWS]	Specifies layout of images on the each page.
Graphics	Group [RWS]	Specifies scaling/conversion options for images.
Labels	Group [RWS]	Specifies appearance of text labels for images on the pages.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Operations::NewDocumentFromImages

Allows to set up of new PDF-page.

Contained Objects

Name	Type	Description
PaperWidth	Double [RWS]	Specifies new PDF page width in points. Default value is **595 pt** (210 mm).
PaperHeight	Double [RWS]	Specifies new PDF page height in points. Default value is **842 pt** (297 mm).
PaperMode	Integer/ String [RWS]	Specifies what paper size to use. Can use custom sizes from **PaperWidth** and **PaperHeight** parameters or standard paper type specified by **PaperName**. For possible named values, see the Paper Modes.

		Default value is **1** (standard).
PaperName	String [RWS]	The name of the standard type of paper. Default value is **A4**.
Orientation	Integer/ String [RWS]	Specifies landscape paper orientation for new document. For possible named values, see the Booleans. Default value is **0**.
MarginLeft	Double [RWS]	Specifies the page left margin in points.
MarginTop	Double [RWS]	Specifies the page top margin in points.
MarginRight	Double [RWS]	Specifies the page right margin in points.
MarginBottom	Double [RWS]	Specifies the page bottom margin in points.

Specifies the layout of images on the each PDF-page.

Contained Objects

Name	Type	Description
HorCount	Integer [RWS]	Count columns on the page. Default value is **1**.
HorSpace	Double [RWS]	Space between columns, in points. Default value is **5 mm** (14,2 pt).
VerCount	Integer [RWS]	Count rows on the page. Default value is **1**.
VerSpace	Double [RWS]	Space between rows, in points. Default value is **5 mm** (14,2 pt).
HorAlign	Integer/ String [RWS]	Aligning in the column. See the Image Align Types. Default value is **"Center"** (1).
VerAlign	Integer/ String [RWS]	Aligning in the row. See the Image Align Types. Default value is **"Center"** (1).

Specifies scaling/conversion options of images.

Contained Objects

Name	Type	Description
ColorImg	Group [RWS]	Scaling/conversion options for colored images.
IndexedImg	Group [RWS]	Scaling/conversion options for indexed images.
MonoImg	Group [RWS]	Scaling/conversation options for monochrome images.
ColorDithering	Integer/ String [RWS]	Enables dithering for colored images when conversion to black & white or monochrome is specified. For possible named values, see the Booleans.

		Default value is **1**.
IndexedDithering	Integer/ String [RWS]	Enables dithering for indexed images when conversion to black & white or monochrome is specified. For possible named values, see the Booleans. Default value is **1**.
MonoThreshold	Integer [RWS]	Specifies threshold for monochrome images when conversion to black & white is specified. Should be between **1** and **255** (boundaries included). Default value is **128**.

Scaling/Conversion options

Name	Type	Description
ScaleType	Integer/ String [RWS]	Specifies image scale type. See the Image Scale Types. Default value is **"None"**(0).
ConvType	Integer/ String [RWS]	Specifies image conversion type. See the Image Conversion Types. Default value is **"None"**(0).
ScaleTo	Integer [RWS]	Specifies dpi of the scaled images. Default value is **96**.
ScaleAbove	Integer [RWS]	If dpi of image is more than it then image will be scaled to the new dpi which is specified in the **ScaleTo**. Default value is **128**.

Remarks

For example (in pseudocode):

```
...
// setup scale type for colored images (enable downsampling)
SetProperty("Operations.NewDocument.FromImages.Graphics.ColorImg.ScaleType
                  DataIn("Bicubic"), NULL, 0);
...
```

Specifies appearance of text labels for images on the pages.

Contained Objects

Name	Type	Description
UseLabels	Integer/ String [RWS]	Enables labels for images. For possible named values, see the Booleans. Default value is **0**.
Text	String [RWS]	Text of label(s). It supports macros for generic names. See Name Generation Macros. The default value is empty.
BackColor	Integer/ String [RWS]	Specifies the background color for text label. For possible named values, see the Colors. Default value is **-1** (none, transparent).
LabelAbove	Integer/ String [RWS]	Moves the text label above image. For possible named values, see the Booleans. Default value is **0**.

2.2.2.11.5.3 FromText

Allows to set settings for creating new PDF document from plain text.

Contained Objects

Name	Type	Description
Paper	Group [RWS]	Specifies new PDF-page options.
Layout	Group [RWS]	Specifies layout of images on the each page.
FileHeader	Group [RWS]	Specifies format of text labels for images on the pages.
NewParaMode	Integer/ String [RWS]	Specifies mode for detecting paragraphs. For possible named values, see the New Paragraph Modes. Default value is **0** (auto-detect).
FilePlaceMode	Integer/ String [RWS]	Specifies mode for placing files into the document. For possible named values, see the Text File Placing Modes. Default value is **0** (new file begins from new paragraph).
CodePage	Integer [RWS]	Specifies the standard code-page identifier of the input text. Default value is **0** (auto-detect).

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Operations::NewDocumentFromText

Allows to set up of new PDF-page.

Contained Objects

Name	Type	Description
PaperWidth	Double [RWS]	Specifies new PDF page width in points. Default value is **595 pt** (210 mm).
PaperHeight	Double [RWS]	Specifies new PDF page height in points. Default value is **842 pt** (297 mm).
PaperMode	Integer/ String [RWS]	Specifies what paper size to use. Can use custom sizes from **PaperWidth** and **PaperHeight** parameters or standard paper type specified by **PaperName**. For possible named values, see the Paper Modes. Default value is **1** (standard).
PaperName	String [RWS]	The name of the standard type of paper. Default value is **A4**.
Orientation	Integer/ String [RWS]	Specifies landscape paper orientation for new document. For possible named values, see the Booleans.

		Default value is **0**.
MarginLeft	Double [RWS]	Specifies the page left margin in points.
MarginTop	Double [RWS]	Specifies the page top margin in points.
MarginRight	Double [RWS]	Specifies the page right margin in points.
MarginBottom	Double [RWS]	Specifies the page bottom margin in points.

Specifies the layout of text on the each PDF-page.

Contained Objects

Name	Type	Description
ColCount	Integer [RWS]	Count columns on the page. Default value is **1**.
ColSpace	Double [RWS]	Space between columns, in points. Default value is **5 pt**.
UseColSep	Integer/ String [RWS]	Enables column separator. For possible named values, see the Booleans. Default value is **0**.
ColSep	Group [RWS]	Defines column separator style.
ColSepColor	Integer/ String [RWS]	Specifies column separator color. For possible named values, see the Colors. Default value is **RGB(0,0,0)** (black).

Defines column separator style.

Contained Objects

Name	Type	Description
Width	Double [RWS]	Separator line thickness in points. Default value is **1 pt**.
BType	String [RWS]	Line type. Supported values: **"S" -** solid**,** **"D" -** dashed. Default value is **"S"**.
DashType	Integer/ String [RWS]	Specifies line dash type. Used if **BType** is equal to **"D"**. For possible named values, see the Dash Types.

Specifies appearance of header which can be placed before each text file.

Contained Objects

Name	Type	Description
UseText	Integer/ String [RWS]	Displays the text label in the header. For possible named values, see the Booleans. Default value is **0**.

Text	String [RWS]	Text of header(s). It supports macros for generic names. See Name Generation Macros. The default value is empty.
BackColor	Integer/ String [RWS]	Specifies the background color for text label. For possible named values, see the Colors. Default value is **-1** (none, transparent).
UseLine	Integer/ String [RWS]	Displays the line in the header. For possible named values, see the Booleans. Default value is **0**.
Line	Group [RWS]	Defines line style.
LineColor	Integer/ String [RWS]	Specifies line color. For possible named values, see the Colors. Default value is **RGB(0,0,0)** (black).
LineAbove	Integer/ String [RWS]	Moves the line above the text label in the header. For possible named values, see the Booleans. Default value is **0**.

Defines line style in file-header.

Contained Objects

Name	Type	Description
Width	Double [RWS]	Line thickness in points. Default value is **1 pt**.
BType	String [RWS]	Line type. Supported values: **"S" -** solid**,** **"D" -** dashed. Default value is **"S"**.
DashType	Integer/ String [RWS]	Specifies line dash type. Used if **BType** is equal to **"D"**. For possible named values, see the Dash Types.

2.2.2.11.6 RotatePages

Allows to specify settings for page rotation.

Contained Objects

Name	Type	Description
RangeType	Integer/ String [RW]	Defines the type of page range for rotation. For possible values, see the Range Types. Supports only **All**, **Selected**, **Current** and **Exact** values. Default value is **1** (all pages).
RangeFilter	Integer/ String [RW]	Specifies an additional filter for the page range. For possible values, see the Range Filters. Default value is **1** (all pages).
RangeText	String [RW]	Specifies the exact pages for rotation. The string should contain the page numbers and/or page ranges separated by commas counting from the start of the document. For example: **"1, 3, 5-12"**.
Direction	Integer/	Determines how to rotate the pages.

	String [RWS]	For possible values, see Rotation Direction. Default value is **0** (Clockwise).

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Operations::RotateDocumentPages, Objects::Documents::<Item>::RotatePages

2.2.2.11.7 SummarizeAnnots

Allows to get a summary all the comments associated with a PDF to save/view them as a new PDF Document, Rich Text Format, Plain Text or Web Page.

Contained Objects

Name	Type	Description
GroupBy	Integer/ String [RWS]	Defines the parameters by which the comments can be grouped. For possible values, see Summarize Annotations Group Types. Default value is **0** (group by page).
SortBy	Integer/ String [RWS]	Defines the parameters by which the comments can be sorted within a group. For possible values, see Summarize Annotations Group Types. Default value is **0** (sort by page).
IncludeInvisibleAnnots	Integer/ String [RWS]	When specified, the invisible comments are included into the summary. For possible named values, see the Booleans. Default value is **0**.
ExcludeReplies	Integer/ String [RWS]	When specified, the replies to the comments are not included into the summary. For possible named values, see the Booleans. Default value is **0**.
SkipUndefinedAnnots	Integer/ String [RWS]	When specified, the undefined comments are not included into the summary. Undefined comment is comment without information which needed for specified grouping/sorting modes. For possible named values, see the Booleans. Default value is **0**.
RangeType	Integer/ String [RWS]	Allows to select the page(s) with the comments to be summarized. For possible values, see RangeTypes. Default value is **1** (all).
RangeFilter	Integer/ String [RWS]	Specifies range filter. For possible values, see RangeFilters. Default value is **1** (all).
RangeText	String [RWS]	Specifies the exact pages to summarize. The string should contain the page numbers and/or page ranges separated by commas counting from the start of the document. For example: **"1, 3, 5-12"**.
RangeReverse	Integer/ String [RW]	Specifies whether to use selected pages in reverse order. For possible named values, see the Booleans. Default value is **0**.

Output	Group [RWS]	Allows to chose output format and settings for it.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Operations::SummarizeDocumentAnnots, Objects::Documents::<Item>::SummarizeAnnots

2.2.2.11.7.1 Output

Allows to chose output format and settings for it.

Contained Objects

Name	Type	Description
Type	Integer/ String [RWS]	Specifies output file type. For possible named values, see Summarize Annotation Output Types.
AutoView	Integer/ String [RWS]	View new document after creation. For possible named values, see the Booleans. Default value is **1**.
ShowPageNumber	Integer/ String [RWS]	Allows to add page number for annotation in output. For possible named values, see the Booleans. Default value is **1**.
ShowTypeName	Integer/ String [RWS]	Allows to add annotation type information in output. For possible named values, see the Booleans. Default value is **0**.
ShowTypeIcon	Integer/ String [RWS]	Allows to add annotation type icon information in output. For possible named values, see the Booleans. Default value is **1**.
ShowSubject	Integer/ String [RWS]	Allows to add subject information for annotation in output. For possible named values, see the Booleans. Default value is **1**.
ShowAuthor	Integer/ String [RWS]	Allows to add author information for annotation. For possible named values, see the Booleans. Default value is **1**.
ShowDate	Integer/ String [RWS]	Allows to add date information for annotation in output. For possible named values, see the Booleans. Default value is **1**.
PDF	Group [RWS]	Specifies output settings for PDF file format.
RTF	Group [RWS]	Specifies output settings for RTF file format.
TXT	Group [RWS]	Specifies output settings for TXT file format.
HTML	Group [RWS]	Specifies output settings for HTML file format.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Objects::Operations.SummarizeAnnots

Specifies output settings for PDF file format.

Contained Objects

Name	Type	Description
LayoutType	Integer/ String [RWS]	Specifies layout type for new PDF document. For possible named values, see Summarize Annotations PDF Layouts.
SkipPagesWithoutAnnots	Integer/ String [RWS]	Do not process PDF pages without annotations. For possible named values, see the Booleans. Default value is **1**.
NewPageForEachGroup	Integer/ String [RWS]	Create new page for each annotation group. For possible named values, see the Booleans. Default value is **1**.
PageWidth	Double [RWS]	Specifies output PDF page width in points. Default value is **595 pt** (210 mm).
PageHeight	Double [RWS]	Specifies output PDF page height in points. Default value is **842 pt** (297 mm).
SaveToFile	Integer/ String [RWS]	When specified, output PDF file will be saved to disk. For possible named values, see the Booleans. Default value is **0**.
FolderName	String [RWS]	Specifies destination folder name. If this value is empty then the <My Documents> standard folder will be used automatically. Default value is empty (NULL or "").
FileName	String [RWS]	Specifies destination file name. It supports macros for generic names. See Name Generation Macros.

See Also

Objects::Operations.SummarizeAnnots.Output

Specifies output settings for RTF file format.

Contained Objects

Name	Type	Description
PageWidth	Double [RWS]	Specifies output RTF page width in points. Default value is **595 pt** (210 mm).
PageHeight	Double [RWS]	Specifies output PDF page height in points. Default value is **842 pt** (297 mm).
NewPageForEachGroup	Integer/ String [RWS]	Create new page for each annotation group. For possible named values, see the Booleans. Default value is **1**.
FolderName	String [RWS]	Specifies destination folder name. If this value is empty then the <My Documents> standard folder

		will be used automatically. Default value is empty (NULL or "").
FileName	String [RWS]	Specifies destination file name. It supports macros for generic names. See Name Generation Macros.

See Also

Objects::Operations.SummarizeAnnots.Output

Specifies output settings for *.txt file.

Contained Objects

Name	Type	Description
FolderName	String [RWS]	Specifies destination folder name. If this value is empty then the <My Documents> standard folder will be used automatically. Default value is empty (NULL or "").
FileName	String [RWS]	Specifies destination file name. It supports macros for generic names. See Name Generation Macros.
CodePage	Integer/ String [RWS]	Specifies code page for output text file. For some possible named values, see Code Pages. Also, you may specify any other code page identifier which supported in system.

See Also

Objects::Operations.SummarizeAnnots.Output

Specifies output settings for HTML file format.

Contained Objects

Name	Type	Description
FolderName	String [RWS]	Specifies destination folder name. If this value is empty then the <My Documents> standard folder will be used automatically. Default value is empty (NULL or "").
FileName	String [RWS]	Specifies destination file name. It supports macros for generic names. See Name Generation Macros.

See Also

Objects::Operations.SummarizeAnnots.Output

2.2.2.12 PageDisplay

Controls document-level page display.

Contained Objects

Name	Type	Description

ArtBoxColor	Integer/ String [RWS]	Specifies the color of the **Art Box**. For possible named values, see the Colors. Default value is **RGB(255,0,0)** (red).
BleedBoxColor	Integer/ String [RWS]	Specifies the color of the **Bleed Box**. For possible named values, see the Colors. Default value is **RGB(0,0,255)** (blue).
BoxesAlpha	Double [RWS]	Specifies the alpha placing coefficient (transparency) for all page-boxes at rendering time. Default value is **1.0** (100% opacity).
CreateLinksFromURLs	Integer/ String [RWS]	Specifies whether to automatically create URLs from text content. For possible named values, see the Booleans. Default value is **1**.
DefaultLayout	Integer/ String [RWS]	Specifies the page layout in the pages view window. For possible values, see the Pages Layouts. Default value is **-1** (auto).
DefaultRotation	Integer [RWS]	Specifies page rotation angle for opening documents. Does not affect on already opened documents. Can be set to 0, 90, 180, 270, 360.
DefaultView	Integer/ String [RWS]	Specifies the default view of the document. For possible values, see the Document Initial View Modes. Default value is **1** (pages view only).
DefaultZoom	Double/ String [RWS]	Specifies the default zoom value for the page view. For possible values, see the Pages Magnifications. Default value is **0** (auto).
Resolution	Integer [RWS]	Specifies the resolution for page display. Default value is **0** (auto, using the resolution of the current monitor).
ShowBoxes	Integer/ String [RWS]	Allows display of **Art**, **Trim**, or **Bleed** boxes. For possible named values, see the Booleans. Default value is **0**.
ShowLargeImages	Integer/ String [RWS]	Allows display of large images. For possible named values, see the Booleans. Default value is **1**.
ShowTransparencyGrid	Integer/ String [RWS]	Allows display of the transparency grid. For possible named values, see the Booleans. Default value is **0**.
SmoothImages	Integer/ String [RWS]	Specifies whether to smooth images. For possible named values, see the Booleans. Default value is **1**.
SmoothLineArt	Integer/ String [RWS]	Specifies whether to smooth line art images. For possible named values, see the Booleans. Default value is **1**.
SmoothText	Integer/ String [RWS]	Specifies whether to smooth text. For possible named values, see the Booleans. Default value is **1**.
TrimBoxColor	Integer/ String [RWS]	Specifies the color of the **Trim Box**. For possible named values, see the Colors. Default value is **RGB(0,255,0)** (green).

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.13 Performance

Controls the performance of the control.

Contained Objects

Name	Type	Description
MaxThreads	Integer [RWS]	Contains maximum number of all special working threads. Default value is **0** (automatic mode).
MemUsage	Integer/ String [RWS]	Represents the number of percents of actual RAM which will be used by control at working in peak. Possible string values, for example: "10%", "25%", "50%", etc. Default value is **0** (automatic mode).
RenderThreads	Integer [RWS]	Contains maximum number of special threads for rendering of pages contents in all opened documents. Default value is **0** (automatic mode).
ThumbThreads	Integer [RWS]	Contains maximum number of special threads for pages thumbnails creation in all opened documents. Default value is **0** (automatic mode).
SyncRendering	Integer/ String [RWS]	Allows to enable the synchronous mode of PDF-content rendering. For possible named values, see the Booleans. Default value is **0** (asynchronous rendering).
UseMemExact	Integer/ String [RWS]	Specifies to use the **MemExact** property. For possible named values, see the Booleans. Default value is **0** (control will use the **MemUsage**).
MemExact	Integer [RWS]	Contains maximum number megabytes (MB) of RAM which can be used by control in peak. Default value is **250** (MB).

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.14 Print

Contains all supported properties for document printing. See also IPDFXCview::PrintDocument.

Contained Objects

Name	Type	Description
AsGrayscale	Integer/String [RWS]	Causes all color information to be converted to grayscale image during printing. For possible named values, see the Booleans. Default value is **0**.
AsImages	Integer/String [RWS]	Converts all pages to images when printing. For possible named values, see the Booleans. Default value is **0**.
AsImagesMaxResol	Integer [RWS]	Sets or gets the maximum page resolution in pixels per inch (DPI) for Print as Images. Valid values are 50, 72, 96, 100, 150, 300, 400, 600 DPI. Default value is **300**.

AllowToFile	Integer [RWS]	Enables/disables the **ToFile** (see below) feature for end-users. For possible named values, see the Booleans. Default value is **1**.
Collate	Integer/String [RW]	Sorts pages during printing. For possible named values, see the Booleans. Default value is **0**.
Copies	String [RW]	Contains number of copies to print. Default value is **1**.
DestFileFolder	String [RWS]	Specifies destination folder name. If this value is empty then the <My Documents> standard folder will be used automatically. Default value is empty (NULL or "").
DestFileTitle	String [RWS]	Specifies destination file name. It supports macros for generic names. See Name Generation Macros.
Duplex	Integer/String [RW]	Selects duplex or double-sided printing for printers capable of duplex printing. For possible named values, see the Duplex Printing. Default value is **0** (None).
GradientsResol	Integer [RWS]	Sets or gets the page resolution in pixels per inch (DPI) for Gradient Fills. Valid values are 50, 72, 96, 100, 150, 300, 400, 600 DPI. Default value is **150**.
NotesAndPopups	Integer/String [RWS]	Prints the PDF document with Notes and Popups. For possible named values, see the Booleans. Default value is **0**.
PaperHeight	Integer [RW]	Paper height in tenths of a millimeter. Default value is **2970**.
PaperName	String [RW]	The name of the standard type of paper to print on.
PaperRotate	Integer [RWS]	Defines the rotation factor for the paper. For possible named values, see the Paper Orientation. Default value is **0**.
PaperWidth	Integer [RW]	Paper width in tenths of a millimeter. Default value is **2100**.
PopupsAlpha	Double [RWS]	Specifies the degree of opacity for Popups. Default value is **0.75**.
PrinterName	String [RW]	Contains name of current printer.
PrintSpec	Integer/String [RWS]	Defines the filter for printing of special PDF content (comments and markups). For possible values, see the Print Specials. By default, document and markups will be printed.
RangeFilter	Integer/String [RWS]	Specifies an additional filter for the page range. For possible values, see the Range Filters. Default value is **1** (all pages).
RangeReverse	Integer/String [RWS]	Specifies whether to print selected pages in reverse order. For possible named values, see the Booleans. Default value is **0**.
RangeText	String [RWS]	Specifies the exact pages to print. The string should contain the page numbers and/or page ranges separated by commas counting from the start of the

		document. For example: **"1, 3, 5-12"**.
RangeType	Integer/String [RWS]	Defines the type of page range. For possible values, see the Range Types. Default value is **1** (all pages).
ScaleSimple	Group [RWS]	Represents advanced options for some simple page scaling during printing for one page to one sheet.
ScaleType	Integer/String [RWS]	Type of scaling to apply during printing. For possible values, see the Print Scale Types. Default value is **0**.
TextAsCurves	Integer/String [RWS]	Allows the text to be printed as curves. For possible values, see the Print Text as Curves. Default value is **1** (ForEmbeddedFonts).
ToFile	Integer/String [RW]	Specifies printing to file. For possible named values, see the Booleans. Default value is **0**.

See Also

IPDFXCview::PrintDocument, IPDFXCview::DoVerb,
IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.14.1 ScaleSimple

Represents advanced options for some simple print scale types such as one PDF page to one sheet of paper. See also Print Scale Types.

Contained Objects

Name	Type	Description
AutoRotate	Integer/ String [RWS]	Auto-rotate pages to match selected paper orientation during printing. For possible values, see the Booleans. Default value is **0**.
AutoCentre	Integer/ String [RWS]	Auto-center pages to match selected paper type during printing. For possible values, see the Booleans. Default value is **0**.
PaperByPage	Integer/ String [RW]	If selected then the appropriate paper will be chosen from list of available papers. Print margins are ignored. For possible values, see the Booleans. Default value is **0**.

See Also

Print Scale Types, IPDFXCview::PrintDocument

2.2.2.15 Prompts

Controls and overrides UI confirmations, prompts, and messages. To receive all confirmations and prompt notifications you need to implement a special event handler (see _IPDFXCviewEvents::OnEvent) and match the types of all received events with the PXCVA_EventTypes::PXCVA_OnDisplayPrompt constants.

Contained Objects

Name	Type	Description
ConfirmDocumentIncSave	Group [RW]	Is raised before saving if document contains the digital signature.
ConfirmDocumentSave	Group [RW]	Is raised if document was changed, before closing.
ConfirmDropFile	Group [RW]	Is raised when detected opening file by drag and drop.
ConfirmFileReplace	Group [RW]	Is raised if destination file already exists.
ConfirmOpenSite	Group [RW]	Is raised when detected click on URL.
CreateFolderError	Group [RW]	Is raised if an error occurs during creation of a new folder appearance.
EnterDocumentPassword	Group [RW]	Is raised during opening of the document if is protected by a password.
FileWriteError	Group [RW]	Is raised if file writing error appears.

Remarks

For all specified objects above you can override the behaviour and appearance of standard UI notifications.
For example (in pseudocode):

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
 ...
    if (Type == PXCVA_OnDisplayPrompt)
    {
       if (Name == "Prompts.ConfirmFileReplace")
       {
          // modify dialog caption
          SetProperty("Prompts.ConfirmFileReplace.UI.Title",
                      DataIn("Sample Title"), 0);
          // modify text of first label
          SetProperty("Prompts.ConfirmFileReplace.UI.Labels[0].Text",
                      DataIn("Sample Text"), 0);
          // skip original dialog display
          // and deny file replace
          SetProperty("Prompts.ConfirmFileReplace.UserChoice",
                      DataIn("No"), 0);
        ...
       }
    }
  ...
}
...
```

Also you may replace the original prompt dialog by your own. To do this you should display your custom dialog in the matching _IPDFXCviewEvents::OnEvent call and send the dialog result to the **UserChoice** property (see User Choices, may not be **0**).

See Also

_IPDFXCviewEvents::OnEvent, PXCVA_EventTypes::PXCVA_OnDisplayPrompt, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.15.1 ConfirmDocumentIncSave

This is a special object for modifying the confirmation dialog when the digitally signed document was changed with incrementally saved changes before closing. See also PXCVA_DocumentSaveFlags::

PXCVA_DocumentSaveInc).

You must implement a special event handler (see _IPDFXCviewEvents::OnEvent) and match the type of the received event with the PXCVA_EventTypes::PXCVA_OnDisplayPrompt constants and the name with the **"Prompts.ConfirmDocumentIncSave"** string.

Contained Objects

Name	Type	Description
ID	Integer [R]	Specifies the unique ID of the document.
FileName	String [RW]	Represents destination file name for specified document.
UserChoice	Integer/ String [RW]	User's choice for this case. Initial value is **0** (default processing). Supported named values for this prompt: **"Default"**, **"Yes"**, **"No"**, **"Cancel"**. See also User Choices.
UI	Group [RW]	Special object for UI customizing.

Remarks

If you want to skip the original dialog you must pass a new non-zero value to the **UserChoice** property. You may also change the destination file name at run time.

For example (in pseudocode):

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
  ...
     if (Type == PXCVA_OnDisplayPrompt)
     {
        if (Name == "Prompts.ConfirmDocumentIncSave")
        {
           // set another file name
           SetProperty("Prompts.ConfirmDocumentIncSave.FileName",
                       DataIn("C:\Test2.pdf"), 0);
           //
           // skip original dialog and
           // try to save document into "C:\Test2.pdf"
           SetProperty("Prompts.ConfirmDocumentIncSave.UserChoice",
                      DataIn("Yes"), 0);
          ...
        }
     }
  ...
}
...
```

See Also

_IPDFXCviewEvents::OnEvent,
PXCVA_EventTypes::PXCVA_OnDisplayPrompt,
IPDFXCview::SaveDocument, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.15.2 ConfirmDocumentSave

This is a special object for modifying the confirmation dialog when a document was changed to save changes before closing.

You must implement a special event handler (see _IPDFXCviewEvents::OnEvent) and match the type of the received event with the PXCVA_EventTypes::PXCVA_OnDisplayPrompt constants and the name with the **"Prompts.ConfirmDocumentSave"** string.

Contained Objects

Name	Type	Description
ID	Integer [R]	Specifies the unique ID of the document.
FileName	String [R]	Represents destination file name for specified document.
UserChoice	Integer/ String [RW]	User's choice for this case. Initial value is **0** (default processing). Supported named values for this prompt: **“Default”**, **“Yes”**, **“No”**, **“Cancel”**. See also User Choices.
UI	Group [RW]	Special object for UI customizing.

Remarks

If you want to skip the original dialog you must pass a new non-zero value to the **UserChoice** property. You may also change the destination file name at run time.

For example (in pseudocode):

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
  ...
     if (Type == PXCVA_OnDisplayPrompt)
     {
        if (Name == "Prompts.ConfirmDocumentSave")
        {
            // set another file name
            SetProperty("Prompts.ConfirmDocumentSave.FileName",
                        DataIn("C:\Test2.pdf"), 0);
            //
            // skip original dialog and
            // try to save document into "C:\Test2.pdf"
            SetProperty("Prompts.ConfirmDocumentSave.UserChoice",
                        DataIn("Yes"), 0);
          ...
        }
     }
  ...
}
...
```

See Also

_IPDFXCviewEvents::OnEvent,
PXCVA_EventTypes::PXCVA_OnDisplayPrompt,
IPDFXCview::SaveDocument, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.15.3 ConfirmDropFile

This is a special object for modifying the file drag and drop operation.

You must implement a special event handler (see _IPDFXCviewEvents::OnEvent) and match the type of

the received event with the PXCVA_EventTypes::PXCVA_OnDisplayPrompt constants and the name with the **"Prompts.ConfirmDropFile"** string.

Contained Objects

Name	Type	Description
FileName	String [RW]	Represents path to file which was dropped in to the Viewer ActiveX.
UserChoice	Integer/ String [RW]	User's choice for this case. Initial value is **1** (default processing). Supported named values for this prompt: **“OK”**, **“Cancel”**, **“Abort”**, **“Ignore”**, **“Yes”**, **“No”**, **“YesToAll”**, **“NoToAll”**. See also User Choices.

Remarks

You may also change the dropped file name at the run time.
Drag and Drop may be disabled/enabled.
For more information, see Objects::General.DenyOpenDocumentsWhenDrop.

For example (in pseudocode):

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
 ...
    if (Type == PXCVA_OnDisplayPrompt)
    {
       if (Name == "Prompts.ConfirmDropFile")
       {
          // get current file path
          GetProperty("Prompts.ConfirmDropFile.FileName",
                      DataOut, 0);

          if (DataOut == "c:\unknownfile.pdf")
          {
            // change file path
            SetProperty("Prompts.ConfirmDropFile.FileName",
                        DataIn("c:\MyFile.pdf"), 0);
          }
          else // ignore
          if (DataOut == "c:\ignorefile.pdf")
          {
            // ignore opening
            SetProperty("Prompts.ConfirmDropFile.UserChoice",
                        DataIn("No"), 0);
          }
         ...
       }
    }
 ...
}
...
```

See Also

_IPDFXCviewEvents::OnEvent,
PXCVA_EventTypes::PXCVA_OnDisplayPrompt,
IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.15.4 ConfirmFileReplace

This is a special object for modifying the confirmation dialog when an error occurs when the specified destination file already exists.

You must implement a special event handler (see _IPDFXCviewEvents::OnEvent) and match the type of the received event with the PXCVA_EventTypes::PXCVA_OnDisplayPrompt constants and the name with the **"Prompts.ConfirmFileReplace"** string.

Contained Objects

Name	Type	Description
FileName	String [RW]	Represents full file name.
UserChoice	Integer/ String [RW]	User's choice for this case. Initial value is **0** (default processing). Supported named values for this prompt: **“Default”**, **“Yes”**, **“YesToAll”**, **“No”**, **“NoToAll”**, **“Cancel”**, **“Automatic”**, **“SaveAs”**. See also User Choices.
UI	Group [RW]	Special object for UI customizing.

Remarks

If you want to skip the original dialog you must pass a new non-zero value to the **UserChoice** property. You may also change the destination file name at run time.

For example (in pseudocode):

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
 ...
    if (Type == PXCVA_OnDisplayPrompt)
    {
       if (Name == "Prompts.ConfirmFileReplace")
       {
          // set another file name
          SetProperty("Prompts.ConfirmFileReplace.FileName",
                      DataIn("C:\Test2.pdf"), 0);
          //
          // skip original dialog and
          // try to save document into "C:\Test2.pdf"
          SetProperty("Prompts.ConfirmFileReplace.UserChoice",
                      DataIn("Yes"), 0);
         ...
       }
    }
 ...
}
...
```

See Also

_IPDFXCviewEvents::OnEvent,
PXCVA_EventTypes::PXCVA_OnDisplayPrompt,
IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.15.5 ConfirmLaunchFile

This is a special object for modifying the confirmation dialog when detected click on file link in PDF document.

You must implement a special event handler (see _IPDFXCviewEvents::OnEvent) and match the type of the received event with the PXCVA_EventTypes::PXCVA_OnDisplayPrompt constants and the name with the **"Prompts.ConfirmLaunchFile"** string.

Contained Objects

Name	Type	Description
FileName	String [RW]	Represents file name. Can be replaced by other file name on processing this event.
UserChoice	Integer/ String [RW]	User's choice for this case. Initial value is **0** (default processing). Supported named values for this prompt: **“Default”**, **“Yes”**, **“OK”**, **“No”**, **“Cancel”**. See also User Choices.
Action	String [RW]	Specifies the action to be performed. Possible values is "open" or "print".
DefDirectory	String [RW]	Specifies the default (working) directory for the action.
Parameters	String [RW]	Specifies the parameters to be passed to the application.

Remarks

If you want to skip the original dialog you must pass a new non-zero value to the **UserChoice** property. You may also change the FileName at run time.

For example (in pseudocode):

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
 ...
    if (Type == PXCVA_OnDisplayPrompt)
    {
       if (Name == "Prompts.ConfirmLaunchFile")
       {
          // get current url
          GetProperty("Prompts.ConfirmLaunchFile.FileName",
                      DataOut, 0);

          if (DataOut == "c:\my.pdf")
          {
            // set another FileName
            SetProperty("Prompts.ConfirmLaunchFile.FileName",
                        DataIn("c:\myNew.pdf"), 0);
          }
          //
          // skip original dialog and
          // try to open specified file
          SetProperty("Prompts.ConfirmLaunchFile.UserChoice",
                      DataIn("Yes"), 0);
         ...
       }
    }
 ...
```

```
}
...
```

See Also

_IPDFXCviewEvents::OnEvent,
PXCVA_EventTypes::PXCVA_OnDisplayPrompt,
IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.15.6 ConfirmOpenSite

This is a special object for modifying the confirmation dialog when detected click on hyperlink in PDF document.

You must implement a special event handler (see _IPDFXCviewEvents::OnEvent) and match the type of the received event with the PXCVA_EventTypes::PXCVA_OnDisplayPrompt constants and the name with the **"Prompts.ConfirmOpenSite"** string.

Contained Objects

Name	Type	Description
Name	String [RW]	Represents URL. Can be replaced by other URL on processing this event.
UserChoice	Integer/ String [RW]	User's choice for this case. Initial value is **0** (default processing). Supported named values for this prompt: **“Default”**, **“Yes”**, **“OK”**, **“No”**, **“Cancel”**. See also User Choices.
KeepUserChoice	Integer/ String [RW]	Keep current choice for this site. If it means **1** then user's choice will be saved into internal control's database. For possible named values, see the Booleans.

Remarks

If you want to skip the original dialog you must pass a new non-zero value to the **UserChoice** property. You may also change the URL at run time.

For example (in pseudocode):

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
 ...
    if (Type == PXCVA_OnDisplayPrompt)
    {
       if (Name == "Prompts.ConfirmOpenSite")
       {
          // get current url
          GetProperty("Prompts.ConfirmOpenSite.Name",
                      DataOut, 0);

          if (DataOut == "www.unknownsite.com")
          {
            // set another URL
            SetProperty("Prompts.ConfirmOpenSite.Name",
                        DataIn("http://www.MySite.com"), 0);
          }
          //
          // skip original dialog and
          // try to open specified URL
```

```
                SetProperty("Prompts.ConfirmOpenSite.UserChoice",
                            DataIn("Yes"), 0);
              ...
            }
        }
     ...
    }
    ...
```

See Also

_IPDFXCviewEvents::OnEvent,
PXCVA_EventTypes::PXCVA_OnDisplayPrompt,
IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.15.7 CreateFolderError

This is a special object for modifying the confirmation dialog when an error occurs during new folder creation.

You must implement a special event handler (see _IPDFXCviewEvents::OnEvent) and match the type of the received event with the PXCVA_EventTypes::PXCVA_OnDisplayPrompt constants and the name with the **"Prompts.CreateFolderError"** string.

Contained Objects

Name	Type	Description
FolderName	String [RW]	Represents full folder name.
ErrorCode	Integer [R]	Creation folder error code.
UserChoice	Integer/ String [RW]	User's choice for this case. Initial value is **0** (default processing). Supported named values for this prompt: **“Default”**, **“Retry”**, **“Cancel”**. See also User Choices.
UI	Group [RW]	Special object for UI customizing.

Remarks

If you want to skip the original dialog you must pass a new non-zero value to the **UserChoice** property. You may also change the new folder name at run time.

For example (in pseudocode):

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
 ...
    if (Type == PXCVA_OnDisplayPrompt)
    {
       if (Name == "Prompts.CreateFolderError")
       {
          // get file writing error if need
          GetProperty("Prompts.CreateFolderError.ErrorCode",
                      DataOut, 0);
          // analyse result in DataOut
          // set another folder name
          SetProperty("Prompts.CreateFolderError.FolderName",
                      DataIn("C:\TestDir\"), 0);
          //
```

```
                // skip original dialog and
                // try folder creation for new path
                SetProperty("Prompts.CreateFolderError.UserChoice",
                            DataIn("Retry"), 0);
              ...
            }
        }
      ...
    }
    ...
```

See Also

_IPDFXCviewEvents::OnEvent,
PXCVA_EventTypes::PXCVA_OnDisplayPrompt,
IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.15.8 EnterDocumentPassword

This is a special object for modifying the confirmation dialog when a password-protected document requires a password.

You must implement a special event handler (see _IPDFXCviewEvents::OnEvent) and match the type of the received event with the PXCVA_EventTypes::PXCVA_OnDisplayPrompt constants and the name with the **"Prompts.EnterDocumentPassword"** string.

Contained Objects

Name	Type	Description
FileName	String [R]	Represents source file name of document.
Password	Integer [RW]	Password for opening the document.
UserChoice	Integer/ String [RW]	User's choice for this case. Initial value is **0** (default processing). Supported named values for this prompt: **“Default”**, **“OK”**, **“Cancel”**. See also User Choices.
UI	Group [RW]	Special object for UI customizing.

Remarks

If you want to skip the original dialog you must pass a new non-zero value to the **UserChoice** property.

For example (in pseudocode):

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
  ...
     if (Type == PXCVA_OnDisplayPrompt)
     {
        if (Name == "Prompts.EnterDocumentPassword")
        {
           // get current file name if need
           GetProperty("Prompts.EnterDocumentPassword.FileName",
                       DataOut, 0);
           // set open password
           SetProperty("Prompts.EnterDocumentPassword.Password",
                       DataIn("123"), 0);
```

```
                // and skip original dialog
                SetProperty("Prompts.EnterDocumentPassword.UserChoice",
                          DataIn("OK"), 0);
              ...
            }
        }
      ...
    }
    ...
```

See Also

_IPDFXCviewEvents::OnEvent,
PXCVA_EventTypes::PXCVA_OnDisplayPrompt,
IPDFXCview::OpenDocument, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.15.9 FileWriteError

This is a special object for modifying the confirmation dialog when an error occurs during file writing.

You must implement a special event handler (see _IPDFXCviewEvents::OnEvent) and match the type of the received event with the PXCVA_EventTypes::PXCVA_OnDisplayPrompt constants and the name with the **"Prompts.FileWriteError"** string.

Contained Objects

Name	Type	Description
FileName	String [RW]	Represents destination file name for specified document.
ErrorCode	Integer [R]	File writing error code.
UserChoice	Integer/ String [RW]	User's choice for this case. Initial value is **0** (default processing). Supported named values for this prompt: **“Default”**, **“Abort”**, **“Retry”**, **“Ignore”**, **“SaveAs”**. See also User Choices.
UI	Group [RW]	Special object for UI customizing.

Remarks

If you want to skip the original dialog you must pass a new non-zero value to the **UserChoice** property. You may also change the destination file name at run time.

For example (in pseudocode):

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
  ...
     if (Type == PXCVA_OnDisplayPrompt)
     {
        if (Name == "Prompts.FileWriteError")
        {
           // get file writing error if need
           GetProperty("Prompts.FileWriteError.ErrorCode",
                     DataOut, 0);
           // analyse result in DataOut
           // set another file name
           SetProperty("Prompts.FileWriteError.FileName",
```

```
                    DataIn("C:\Test2.pdf"), 0);
            //
            // skip original dialog and
            // try to save document into "C:\Test2.pdf"
            SetProperty("Prompts.FileWriteError.UserChoice",
                    DataIn("Retry"), 0);
              ...
          }
        }
      ...
    }
    ...
```

See Also

_IPDFXCviewEvents::OnEvent,
PXCVA_EventTypes::PXCVA_OnDisplayPrompt,
IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.16 Search

Contains all supported properties for extended text searching using search panel in Viewer.

Contained Objects

Name	Type	Description
What	Group [RWS]	Specifies text for search.
Where	Group [RWS]	Specifies options for place of search.
Options	Group [RWS]	Specifies options for search.
AllowOpenExternalDoc	Integer/String [RWS]	Allow to open external pdf documents via results list in Search panel. For possible named values, see the Booleans. Default value is **1** (true).
AllowSearchInFolders	Integer/String [RWS]	Allow to search text in folders. Switches on/off the folders tree in the search pane. For possible named values, see the Booleans. Default value is **1** (true).
AllowSearchResBlinking	Integer/String [RWS]	Show blinking selection on the found text. For possible named values, see the Booleans. Default value is **1** (true).

Remarks

To start search you may execute **"StartFullSearch"** UI-command.
To stop search - execute **"StopFullSearch"**.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Commands List

2.2.2.16.1 Options

Contains options for text search.

Contained Objects

Name	Type	Description
WholeWordsOnly	Integer/String [RWS]	Search only whole words. For possible named values, see the Booleans. Default value is **0**.
CaseSensitive	Integer/String [RWS]	Use case sensitive search. For possible named values, see the Booleans. Default value is **0**.
IncludeBookmarks	Integer/String [RWS]	Search in bookmarks. For possible named values, see the Booleans. Default value is **0**.
IncludeComments	Integer/String [RWS]	Search in comments. For possible named values, see the Booleans. Default value is **0**.
LookInSubFolders	Integer/String [RWS]	Search in sub folders if search in folder mode specified. For possible named values, see the Booleans. Default value is **1**.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.16.2 What

Specifies text for searching.

Contained Objects

Name	Type	Description
Text	String [RWS]	Specifies text for search in PDF documents. Text may contains some boolean operators: AND, OR, NOT. For examples: *PDF Reference,* *PDF **AND** Reference* (identical to: *PDF Reference*), *PDF **OR** Reference,* *PDF **NOT** Reference.* If search results should contain some words which are logical operators please quote these words in the input search text, for example: if you want to search the text: *Service is not available* please fix it before to: *Service is **"not"** available*

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.16.3 Where

Specifies setting for destination search place.

Contained Objects

Name	Type	Description

Mode	Integer/String [RWS]	Specifies place to search. For possible named values, see the Search Modes. Default value is **0** (**"InActive"**).
FolderName	String [RW]	Specifies folder path for searching in it. This parameter is used only if **Mode** is set to **" InFolder"**.
Disabled	Integer/String [RWS]	Disables the **[Search Pane/Where]** combo-box. For possible named values, see the Booleans. Default value is **0**.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.17 Tools

Contains all supported tools and operations with them.

Contained Objects

Name	Type	Description
Active	Integer/ String[RWS]	Specifies which tool is active. For possible values, see Tools.
Area	Group [RWS]	This tool allows to measure the area within the line segments.
Arrow	Group [RWS]	This tool allows to draw a arrow on the current displayed PDF document page.
Callout	Group [RWS]	This tool allows to add a text box with an arrow pointing to a selected location on the page.
Cloud	Group [RWS]	This tool draws a "cloud" outline around selected parts of the PDF page.
Distance	Group [RWS]	This tool permits measuring the distance between two points.
Eraser	Group [RWS]	This tool allows to erase any part of a pencil-drawn annotation.
FileAttachment	Group [RWS]	This tool allows to add file-attachment annotation in any place of PDF document.
Highlight	Group [RWS]	This tool permits to highlight text within the PDF document, and works like the select tool.
Line	Group [RWS]	This tool allows to draw a line on the current displayed PDF document page.
Link	Group [RWS]	This tool allows to create a link in order to jump to other locations in the same document or websites.
Loupe	Group [RWS]	This tool opens a special dialog window that is linked to a scrollable, resizable rectangular are that acts as a cursor and permits the selection of a portion of the PDF page/file.
Oval	Group [RWS]	This tool allows you to draw ovals and circles on the currently displayed PDF document page.
Pencil	Group [RWS]	This tool allows to draw a freehand shape on the document.
Perimeter	Group	This tool allows to measure a set of distances

	[RWS]	between multiple points.
Polygon	Group [RWS]	This tool draws a polygon outline around selected parts of the PDF page.
PolyLine	Group [RWS]	This tool is an extension of the basic Line Tool that allows you to draw multiple connected line segments.
Rect	Group [RWS]	This tool allows to draw a rectangle on the current displayed PDF document page.
Snapshot	Group [RWS]	This tool permits copying a rectangular area of a PDF page or the entire visible page area to the clipboard as a Bitmap.
Stamp	Group [RWS]	This tool allows to apply standard stamps to the currently displayed PDF document page.
StickyNote	Group [RWS]	This tool allows to add sticky note in any place of PDF document.
StrikeOut	Group [RWS]	This tool permits to strike out selected text.
TextBox	Group [RWS]	This tool allows to add text on a page in a text box.
TypeWriter	Group [RWS]	This tool allows to enter text comments into PDF page.
Underline	Group [RWS]	This tool permits to underline selected text.

Remarks

Examples for usage (in pseudocode):

1. set line as active tool:

```
...
SetProperty("Tools.Active", "Line");
...
```

2. change fill color for current style of line tool:

```
...
GetProperty("Tools.Line.Style", DataOut);
StyleID = DataOut;
SetProperty("Commenting.Line.Styles[#" + StyleID + "].FColor", "blue");
...
```

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, Objects::Commenting

2.2.2.17.1 Area

Area tool allows to measure the area within the line segments.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Area.Styles.
KeepSelected	Integer/	Defines if this tool still be selected after its

	String [RWS]	usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.2 Arrow

This tool allows to draw a arrow on the current displayed PDF document page.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Arrow.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.3 Callout

The Callout tool allows to add a text box with an arrow pointing to a selected location on the page.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Callout.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.4 Cloud

The Cloud Tool draws a "cloud" outline around selected parts of the PDF page.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool.

		See also: Objects::Commenting.Cloud.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.5 Distance

Distance tool permits measuring the distance between two points.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Distance.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.6 Eraser

This tool allows to erase any part of a pencil-drawn annotation.

Contained Objects

Name	Type	Description
Diameter	Double [RWS]	Defines diameter for the eraser tool. Default value is **10** pt.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools

2.2.2.17.7 FileAttachment

This tool allows to add file attachment in any place of PDF document.

Contained Objects

Name	Type	Description
Style	Integer	Defines ID of current commenting style for this

	[RWS]	tool. See also: Objects::Commenting.FileAttachment.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.8 Highlight

The highlight tool permits to highlight text within the PDF document, and works like the select tool.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Highlight.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.9 Line

This tool allows to draw a line on the current displayed PDF document page.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Line.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.10 Link

Link tool allows to create a link in order to jump to other locations in the same document or websites.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Link.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.11 Loupe

This tool opens a special dialog window that is linked to a scrollable, resizable rectangular are that acts as a cursor and permits the selection of a portion of the PDF page/file.

Contained Objects

Name	Type	Description
ZoomLevel	Double [RS]	Defines zoom level. Default value is **200** percent.

See Also

Objects::Tools

2.2.2.17.12 Oval

The Oval (Circle) tool allows you to draw ovals and circles on the currently displayed PDF document page.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Oval.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.13 Pencil

The Pencil Tool allows to draw a freehand shape on the document.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Pencil.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.14 Perimeter

Perimeter tool allows to measure a set of distances between multiple points.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Perimeter.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.15 Polygon

The Polygon Tool draws a polygon outline around selected parts of the PDF page.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Polygon.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.16 PolyLine

The Polygon Line Tool is an extension of the basic Line Tool that allows you to draw multiple connected line segments.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Polyline.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.17 Rect

Rectangle tool allows to draw a rectangle on the current displayed PDF document page.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Rect.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.18 Snapshot

This tool permits copying a rectangular area of a PDF page or the entire visible page area to the clipboard as a Bitmap.

Contained Objects

Name	Type	Description
UseResolution	Integer/ String [RWS]	If this value sets to 0 then screen resolution for output image will be taken. For possible named values, see the Booleans. Default value is **0**.
Resolution	Integer/ String [RWS]	If UseResolution is set to 1 then this parameter specifies resolution for output image Default value is **72** dpi.

AsGrayscale	Integer/ String [RWS]	Gets or sets grayscale mode for snapshot. For possible named values, see the Booleans. Default value is **0**.
NoSounds	Integer/ String [RWS]	Disable snapshot sound. For possible named values, see the Booleans. Default value is **0**.

See Also

Objects::Tools

2.2.2.17.19 Stamp

This tool allows to apply standard stamps to the currently displayed PDF document page.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Stamp.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.20 StickyNote

This tool allows to add sticky note in any place of PDF document.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.StickyNote.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.21 StrikeOut

The StrikeOut tool permits to cross out selected text.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.StrikeOut.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.22 TextBox

This tool allows to add text on a page in a text box.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.TextBox.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.23 TypeWriter

The typewriter tool allows to enter text comments into PDF page.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.TypeWriter.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.17.24 Underline

The underline tool permits to underline selected text.

Contained Objects

Name	Type	Description
Style	Integer [RWS]	Defines ID of current commenting style for this tool. See also: Objects::Commenting.Underline.Styles.
KeepSelected	Integer/ String [RWS]	Defines if this tool still be selected after its usage. For possible named values, see the Booleans. Default value is **1**.

See Also

Objects::Tools, Objects::Commenting

2.2.2.18 View

This is the object for controlling the main view attributes: visibility, position, layout for main/document bars, panes, windows. It also provides customizing of main menu and toolbars.

Contained Objects

Name	Type	Description
Bars	Array [RWS]	Represents all UI bars (menu, command, status bars). For details about main bars, see the Main Bars.
Colors	Group [RWS]	Defines colors of the control's user interface.
LockAllBars	Integer/ String [RWS]	Deny/allow customization and reposition for all command bars. Default value is **0** (all command bars are unlocked). For possible named values, see Booleans.
Locked	Integer/ String [RWS]	Deny/allow all UI-customization. Default value is **0**. For possible named values, see Booleans.
Panes	Array [RWS]	Represents all UI panes (named containers for special views). For details about main panes, see the Main Panes.
ShowDocumentViewBorder	Integer/ String [RWS]	Enables/disables 3D border for document view area (pages view). For possible named values, see Booleans.
ShowWorkspaceBorder	Integer/ String [RWS]	Enables/disables 3D border for control's workspace area. Note: the effect can only be seen when there is no opened document. For possible named values, see Booleans.
ShowTooltipsOnToolbars	Integer/ String [RWS]	Enables/disables tooltips for all toolbars/menus. Default value is **1**. For possible named values, see Booleans.
Show3DToolbars	Integer/	Enables/disables borders for all main toolbars.

	String [RWS]	Default value is **0**. For possible named values, see Booleans.
ShowSepBetweenToolbars	Integer/ String [RWS]	Enables/disables horizontal 3D-separators between main toolbars. Default value is **1**. For possible named values, see Booleans.
ShowShortcutsInTooltips	Integer/ String [RWS]	Show/hide shortcuts in tooltips. Default value is **1**. For possible named values, see Booleans.
ShowShortcutsInMenus	Integer/ String [RWS]	Show/hide shortcuts in menu items. Default value is **1**. For possible named values, see Booleans.
ShowTabCloseButtons	Integer/ String [RWS]	Show/hide close button in tabs of documens. Default value is **1**. For possible named values, see Booleans.
HideTabBarForSingleDoc	Integer/ String [RWS]	If specified then control will hide documents tab bar if only one document is opened. Default value is **0**. For possible named values, see Booleans.
ShowAppWorkspaceShadow	Integer/ String [RWS]	Enables/disables top/bottom shadow on the main background. Default value is **1**. For possible named values, see Booleans.

Remarks

Examples for usage (in pseudocode):

1.1. get bars count (same for panes):

```
...
DoVerb("View.Bars.Count", "get", DataIn, DataOut, 0);
// or:
GetProperty("View.Bars.Count", DataOut, 0);
...
```

1.2. get bar unique name by order index (same for panes):

```
...
DoVerb("View.Bar[0].Name", "get", DataIn, DataOut, 0);
// or:
GetProperty("View.Bar[0].Name", DataOut, 0);
...
```

1.3. get menu bar visibility:

```
...
DoVerb("View.Bars[\"Menu\"].Visible", "get", DataIn, DataOut, 0);
// or:
GetProperty("View.Bars[\"Menu\"].Visible", DataOut, 0);
...
```

1.4. get full search pane visibility:

```
...
DoVerb("View.Panes["Search"].Visible", "get", DataIn, DataOut, 0);
// or:
GetProperty("View.Panes["Search"].Visible", DataOut, 0);
...
```

2.1. show 'file' toolbar:

```
...
DoVerb("View.Bar["File"].Visible", "set", DataIn(1), DataOut, 0);
// or:
SetProperty("View.Bar["File"].Visible", DataIn(1), 0);
...
```

2.2. show full search pane:

```
...
```

```
DoVerb("View.Panes["Search"].Visible", "set", DataIn(1), DataOut, 0);
// or:
SetProperty("View.Panes["Search"].Visible", DataIn(1), 0);
...
```

See Also

Object Name Notation,
Objects::Documents::<Item>.View

2.2.2.18.1 Colors

Defines colors of the control's user interface.

Contained Objects

Name	Type	Description
Back	Integer/ String [RWS]	Background (shadow) color for three-dimensional display elements. For possible named values, see the Colors.
Face	Integer/ String [RWS]	Face color for three-dimensional display elements. For possible named values, see the Colors.
Highlight	Integer/ String [RWS]	Color for highlighted by selection items. For possible named values, see the Colors.
Mark	Integer/ String [RWS]	Mark color for selected items. For possible named values, see the Colors.
Text	Integer/ String [RWS]	Color for text UI elements. For possible named values, see the Colors.
Window	Integer/ String [RWS]	Background color for window elements. For possible named values, see the Colors.

Next properties controls overriding colors of some UI-elements:

AppWorkspace	Integer/ String [RWS]	Color for the application workspace. Can be seen when no opened document in control or in MDI mode. For possible named values, see the Colors.
PagesBack	Integer/ String [RWS]	Background color of the document's pages view. For possible named values, see the Colors.

Remarks

If color value is set to **-1** then it means that default(system) color for specified UI element will be taken.
Example for usage (in pseudocode):

```
...
SetProperty("View.Colors.PagesBack", RGB(0,0,255), 0);
GetProperty("View.Colors.Face", DataOut, 0);
```

See Also

Objects::View

2.2.2.19 Multiply Referred Objects

These objects are multiply referred from other instances.

Objects

Name	Type	Description
Bars	Array	Represents an array of UI bars. For example, see Objects::View.
Border	Group	Represents border structure for comment. For example: Objects::Commenting.
Panes	Array	Represents an array of UI panes. For example, see Objects::View.
RectangleF	Group	Represents a rectangle structure. For example: Objects::Documents::<Item>.Pages::<Item>.
Text Format	Group	Represents text format structure for comment. For example: Objects::Commenting.
UI	Group	Special object for UI customization. For example, see Objects::Prompts.CreateFolderError.

2.2.2.19.1 Bars

This object is an array of items, each item representing a UI bar (menu, toolbar, status bar, etc.).

Item Template

<Item> - defines item template for each bar representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains the number of bars.

See Also

Objects::View,
Objects::Documents::<Item>.View

2.2.2.19.1.1 <Item>

Defines the item template for each UI bar.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Defines the unique identifier of the bar.
Name	String [RS]	Defines the unique name of the bar.
Visible	Integer/ String [RWS]	Determines the visibility of the bar.

See Also

Objects::View,
Objects::Documents::<Item>.View.

2.2.2.19.2 Border

This object represents border structure for comment.

Contained Objects

Name	Type	Description
Width	Double [RWS]	Specifies the width of the border. Default value is 1.0
Type	Integer/ String [RWS]	Specifies the type of border. For possible named values, see the Border Types. Default value is 0 (Solid)
Effect	Integer/ String [RWS]	Specifies the effect of the border. For possible named values, see the Border Effects. Default value is 0 (None)
DashType	Integer/ String [RWS]	Specifies dash type of the border. For possible named values, see the Dash Types. Default value is 0.
CloudyLevel	Double [RWS]	Specifies cloudy level for border. Default value is 2.0

See Also

Objects::Commenting

2.2.2.19.3 Panes

This object is an array of items, each item representing a UI "pane," or named container for special view window.

Item Template

<Item> - defines item template for each pane representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains the number of panes.

See Also

Objects::View,
Objects::Documents::<Item>.View

2.2.2.19.3.1 <Item>

Defines item template for each UI pane.

Contained Objects

Name	Type	Description
ID	Integer [RS]	Defines the unique identifier of the pane.
Name	String [RS]	Defines the unique name of the pane.
Visible	Integer/ String [RWS]	Determines the visibility of the pane.

See Also

Objects::View,
Objects::Documents::<Item>.View
.

2.2.2.19.4 RectangleF

This object represents rectangle structure.

Contained Objects

Name	Type	Description
Left	Double [RWS]	Specifies the x-coordinate of the upper-left corner of the rectangle.
Top	Double [RWS]	Specifies the y-coordinate of the upper-left corner of the rectangle.
Right	Double [RWS]	Specifies the x-coordinate of the lower-right corner of the rectangle.
Bottom	Double [RWS]	Specifies the y-coordinate of the lower-right corner of the rectangle.

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty

2.2.2.19.5 Text Format

This object represents text format structure for comment.

Contained Objects

Name	Type	Description
Char	Group [RWS]	Specifies character format for text.
Para	Group [RWS]	Specifies paragraph format for the text.

See Also

Objects::Commenting

2.2.2.19.5.1 Char

This object represents char format structure for comment.

Contained Objects

Name	Type	Description
Bold	Integer/ String [RWS]	Specifies text bold style. For possible named values, see the Booleans. Default value is **0** (false).
FColor	Integer/ String [RWS]	Specifies fill color for the text. For possible named values, see the Colors. Default value is **RGB(0,0,0)** (black).
FontName	String [RWS]	Specifies font name. Default value is **Arial**.
FontSize	Double [RWS]	Specifies font size. Default value is **12.0** pt.
HScale	Double [RWS]	Specifies horizontal scale for the text. Default value is **100** percents.
Italic	Integer/ String [RWS]	Specifies text italic style. For possible named values, see the Booleans. Default value is **0** (false).
RMode	Integer/ String [RWS]	Specifies text rendering mode. For possible values, see the Text Rendering Mode. Default value is **0** (Fill).
SColor	Integer/ String [RWS]	Specifies stroke color for the text. For possible named values, see the Colors. Default value is **RGB(0,0,0)** (black).
Strikeout	Integer/ String [RWS]	Specifies text strikeout style. For possible named values, see the Booleans. Default value is **0** (false).
Underline	Integer/ String [RWS]	Specifies text underline style. For possible named values, see the Booleans. Default value is **0** (false).

See Also

Objects::Commenting

2.2.2.19.5.2 Para

This object represents paragraph format structure for comment.

Contained Objects

Name	Type	Description
FirstLineIndent	Double [RWS]	Specifies indent for first line. Default value is **0**.
LineSpacing	Double [RWS]	Specifies line spacing. If value is positive number then its specifies line spacing in percents. If this value is negative - it specifies line spacing in points. Default value is **100** percent.

Margins	Group [RWS]	Specifies paragraph margins.
TextAlign	Integer/ String [RWS]	Specifies text align for paragraph. For possible named values, see TextAlign. Default value is **0** (Left align).

See Also

Objects::Commenting

2.2.2.19.6 UI

This object is used for customizing prompts, confirmations and messages. See also Objects::Prompts.

Contained Objects

Name	Type	Description
Title	String [RW]	Represents the caption text of dialog.
Labels	Array [RW]	Represents all customizable text labels on dialog.

Remarks

To customize prompts, declared in Objects::Prompts section, you must implement a special event handler (see _IPDFXCviewEvents::OnEvent) and match the event(s) needing overrides.

If necessary event was matched, then you can to customize UI: change dialog caption, text labels by object **UI**, *but do not change* the **UserChoice** value in the received prompt object; this value must be **0** to allow the original dialog displaying.

For example (in pseudocode) caption and all text labels for some prompts will be changed:

```
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
    if (Type == PXCVA_OnDisplayPrompt)
    {
        if (Name == "Prompts.CreateFolderError" OR
            Name == "Prompts.ConfirmFileReplace" OR
            Name == "Prompts.ConfirmDocumentSave" OR
            Name == "Prompts.ConfirmDocumentIncSave" OR
            Name == "Prompts.FileWriteError" OR
            Name == "Prompts.EnterDocumentPassword")
        {
            // modify dialog caption
            SetProperty(Name + ".UI.Title",
                        DataIn("Sample Title"), 0);
            // get count of text labels
            GetProperty(Name + ".UI.Labels.Count",
                        DataOut, 0);
            Count = DataOut;
            // modify text of all labels
            for i = 0 to Count-1
            {
                SetProperty(Name + "UI.Labels[" + i + "].Text",
                            DataIn("Sample Text"), 0);
            }
          ...
        }
    }
```

```
}
...
```

See Also

Objects::Prompts,
_IPDFXCviewEvents::OnEvent,
PXCVA_EventTypes::PXCVA_OnDisplayPrompt

2.2.2.19.6.1 Labels

This is a special object for the customization of prompts, confirmations, and messages. It allows changing captions, text labels. See also Objects::Prompts.

Item Template

<Item> - defines item template for each text label representation.

Contained Objects

Name	Type	Description
Count	Integer [R]	Contains the number of all text labels.

See Also

Objects::Prompts

Defines the item template for each text label representation.

Contained Objects

Name	Type	Description
ID	Integer [R]	Defines a special unique identifier of text label.
Text	String [RW]	Contains text of label.

See Also

Objects::Prompts

2.2.2.19.7 Quad

This object represents the coordinates of a quadrilateral area on a PDF page. Coordinates is stored in points.

Contained Objects

Name	Type	Description
Value	Array of Double [R]	Contains an array of numbers(eight) which specify the coordinates of the one quadrilateral area.

Remarks

Coordinates are stored in an array of doubles where each pair specifies the X and Y coordinates

of one point.

Array index	Description
0	X coordinate of **bottom left** point
1	Y coordinate of **bottom left** point
2	X coordinate of **bottom right** point
3	Y coordinate of **bottom right** point
4	X coordinate of **top right** point
5	Y coordinate of **top right** point
6	X coordinate of **top left** point
7	Y coordinate of **top left** point

See Also

Objects::Documents::<Item>.Pages::<Item>.Text.Chars::<Item>, PXCVA_OutArgs

2.2.2.19.8 Quads

This object represents the coordinates of a quadrilateral area(s) on a PDF page. Coordinates is stored in points.

Contained Objects

Name	Type	Description
Value	Array of Double [R]	Contains an array of numbers which specify the coordinates of the quadrilateral area(s).

Remarks

Coordinates are stored in an array of doubles where each pair specifies the X and Y coordinates of one point; each eight doubles(four points) specifies the quadrilateral area.

Array index	Description
0	X coordinate of **bottom left** point of the *first* quadrilateral area
1	Y coordinate of **bottom left** point of the *first* quadrilateral area
2	X coordinate of **bottom right** point of the *first* quadrilateral area
3	Y coordinate of **bottom right** point of the *first* quadrilateral area
4	X coordinate of **top right** point of the *first* quadrilateral area
5	Y coordinate of **top right** point of the *first* quadrilateral area
6	X coordinate of **top left** point of the *first* quadrilateral area
7	Y coordinate of **top left** point of the *first* quadrilateral area
8	X coordinate of **bottom left** point of the *second* quadrilateral area
9	Y coordinate of **bottom left** point of the *second* quadrilateral area
...	

See Also

Objects::Documents::<Item>.Pages::<Item>.Text.Chars::<Item>, PXCVA_OutArgs

2.2.3 Values

This section represents a special named values tables:

- Blend Modes
- Booleans
- Border Effects
- Border Types
- Code Pages
- Color Management Engines
- Colors
- Command States
- Comment Subject Modes
- Content Monitor States
- Context Menu User Choices
- Dash Types
- Document Bars
- Document Initial View Modes
- Document Interfaces
- Document Panes
- Document Save Methods
- Document Save Modes
- Document SaveAs Destination Types
- Duplex Printing
- Export Modes
- Export to Image Modes
- File Attachment Icon Types
- Highlight Form Fields Masks
- Image Align Types
- Image Conversion Types
- Image Scale Types
- Image Types
- Keyboard Notifications Filter Flags
- Line Ending
- Link Highlight Mode
- Main Bars
- Main Panes
- Mouse Notifications Filter Flags
- Name Generation Macros
- New Paragraph Modes
- Pages Layouts
- Pages Magnifications
- Paper Modes
- Paper Orientation
- PDF-Specification Versions
- Print Notifications Filter Flags
- Print Scale Types
- Print Specials
- Print Text as Curves
- Range Filters
- Range Types
- Registry Roots
- Rotation Direction
- Search Modes
- Selection Notifications Filter Flags
- Shortcut Key Types
- Shortcut Modifiers
- Sticky Note Icon Types
- Summarize Annotations Group Types
- Summarize Annotations Output Types
- Summarize Annotations PDF Layouts
- Text Align

- Text Editor Notifications Filter Flags
- Text File Placing Modes
- Text Rendering Mode
- Tools
- User Choices
- User Interface Languages

2.2.3.1 Blend Modes

Supported Values

String Value	Integer Value	Description
Normal	0	Selects the source color, ignoring the backdrop.
Multiply	1	Multiplies the backdrop and source color values.
Screen	2	Multiplies the complements of the backdrop and source color values.
Overlay	3	Multiplies or screens the colors, depending on the backdrop color value. Source colors overlay the backdrop while preserving its highlights and shadows. The backdrop color is not replaced but is mixed with the source color to reflect the lightness or darkness of the backdrop.
Darken	4	Selects the darker of the backdrop and source colors.
Lighten	5	Selects the lighter of the backdrop and source colors.
ColorDodge	6	Brightens the backdrop color to reflect the source color. Painting with black produces no changes.
ColorBurn	7	Darkens the backdrop color to reflect the source color. Painting with white produces no change.
HardLight	8	Multiplies or screens the colors, depending on the source color value. The effect is similar to shining a harsh spotlight on the backdrop.
SoftLight	9	Darkens or lightens the colors, depending on the source color value. The effect is similar to shining a diffused spotlight on the backdrop.
Difference	10	Subtracts the darker of the two constituent colors from the lighter color. Painting with white inverts the backdrop color; painting with black produces no change.
Exclusion	11	Produces an effect similar to that of the **Difference** mode but lower in contrast. Painting with white inverts the backdrop color; painting with black produces no change.
Hue	12	Creates a color with the hue of the source color and the saturation and luminosity of the backdrop color.
Saturation	13	Creates a color with the saturation of the source color and the hue and luminosity of the backdrop color. Painting with this mode in an area of the backdrop that is a pure gray (no saturation) produces no change.
Color	14	Creates a color with the hue and saturation of the source color and the luminosity of the backdrop color. This preserves the gray levels of the backdrop and is useful for coloring monochrome images or tinting color images.
Luminosity	15	Creates a color with the luminosity of the source color and the hue and saturation of the backdrop color. This produces an inverse effect to that of the **Color** mode.

2.2.3.2 Booleans

Supported Values

String Value	Integer Value	Description
True	1	Defines **TRUE** boolean value.
Yes		
Ok		
On		
False	0	Defines **FALSE** boolean value.
No		
Cancel		
Off		

See Also

Named Objects

2.2.3.3 Border Effects

Supported Values

String Value	Integer Value	Description
None	0	No effect.
Cloudy	1	Cloudy line effect.

See Also

Objects::Border

2.2.3.4 Border Types

Supported Values

String Value	Integer Value	Description
Solid	0	Solid line type.
Dashed	1	Dashed line type.

See Also

Objects::Border

2.2.3.5 Code Pages

Supported Values

String Value	Integer Value
Unicode	-2
Dos	-1
Windows	0

UTF8	65001

See Also

Objects::Operations.SummarizeAnnots.Output.TXT

2.2.3.6 Color Management Engines

Supported Values

String Value	Integer Value	Description
LittleCMS	0	**Little CMS** engine will be used for color management
MicrosoftICM	1	**Microsoft ICM** engine will be used for color management.

See Also

Objects::ColorManagement

2.2.3.7 Colors

Supported Values

String Value	Integer Value	Description
White	RGB (255,255,255)	White color
Black	0	Black color
Red	RGB(255,0,0)	Red color
Green	RGB(0,255,0)	Green color
Blue	RGB(0,0,255)	Blue color
Gray	RGB (192,192,192)	Grey color
Grey	RGB (192,192,192)	Grey color

Note: this table can be expanded in future.

See Also

Objects::PageDisplay,
Objects::Export.Image

2.2.3.8 Command States

Supported Values

String Value	Integer Value	Description
Enabled	1	This command is enabled currently and available for execution.
Disabled	0	This command is disabled currently and cannot be executed.
Offline	-1	Turns off the command in control's UI. All UI-locations of the command will be disabled. This state can be removed only by the external application by setting the state into "*Online*" value.
Off		

Online On	2	Turns on the command in control's UI (i.e. to remove "*Offline*" state only).

See Also

Objects::Commands::<Item>

2.2.3.9 Comment Subject Modes

Supported Values

String Value	Integer Value	Description
Default	0	Use default subject.
Custom	1	Use custom subject text which is specified in **DefSubj** property of specific comment style. See, for example: Objects::Commenting.Line.Styles::<Item>.DefSubj
Common	2	Use common subject text which is specified in the Objects::Commenting.AnnotCommonSubj property.

2.2.3.10 Content Monitor States

Supported Values

String Value	Integer Value	Description
Off	0	Disables the **Content Monitor**. The document's property Objects::Documents::<Item>.ContentReady will be undefined and non-actual always.
On	1	Enables the **Content Monitor**. The document's property Objects::Documents::<Item>.ContentReady will be actual. Notification about each change of this property can be received through _IPDFXCviewEvents::OnEvent.
Once	2	Enables the **Content Monitor** for the first change of the Objects::Documents::<Item>.ContentReady only and disables after.

See Also

Objects::Documents.ContentMonitor,
Objects::Documents::<Item>.ContentReady

2.2.3.11 Context Menu User Choices

Supported Values

String Value	Integer Value
Default	0
Deny	-1
Skip	-1

See Also

Objects::Notifications.ContextMenu

2.2.3.12 Dash Types

Supported Values

String Value	Integer Value	Description
Default	1	2 units off, 2 units on.
Dot2		
Dot3	2	3 off, 2 on.
Dot4	3	4 off, 4 on.
DashDot	4	4 off, 3 on, 2 off, 3 on.
DotDash	5	4 off, 3 on, 16 off, 3 on.
DashDot2	6	8 off, 4 on, 4 off, 4 on.

See Also

Objects::Border

2.2.3.13 Document Bars

Supported Document Bars Identifiers

Name	ID	Description
Options	33262	Specifies the document's **Options** toolbar.
PagesNavigation	33263	Specifies the document's **Pages Navigation** toolbar.
PagesLayout	33264	Specifies the document's **Pages Layout** toolbar.
Launch	33265	Specifies the document's **Launch** toolbar.

Remarks

Examples for usage (in pseudocode):

1. get **Options** toolbar visibility:

```
...
GetDocumentProperty(4095, "View.Bars["Options"].Visible", DataOut, 0);
// or:
GetDocumentProperty(4095, "View.Bars[#33262].Visible", DataOut, 0);
...
```

2. show **Options** toolbar:

```
...
SetDocumentProperty(4095, "View.Bars["Options"].Visible", DataIn(1), 0);
// or:
SetDocumentProperty(4095, "View.Bars[#33262].Visible", DataIn(1), 0);
...
```

See Also

Objects::Documents::<Item>.View

2.2.3.14 Document Initial View Modes

Supported Values

String	Integer	Description

Value	Value	
Automatic Auto	-1	Selects view of the document automatically.
PageOnly	1	Displays pages view only.
BookmarksAndPage	3	Displays bookmarks and pages only.
ThumbnailsAndPage	5	Displays thumbnails and pages only.
LastUsedView	0	Displays the last used view of the document.

See Also

Objects::PageDisplay.DefaultView

2.2.3.15 Document Interfaces

Supported Values

String Value	Integer Value	Description
MDI Multiple	0	Multiple-document interface (MDI) displays multiple documents at the same time, with each document displayed in its own window.
SDI Single	1	Single-document interface.

See Also

Objects::General.DocumentInterface

2.2.3.16 Document Panes

Supported Document Panes Identifiers

Name	ID	Description
Bookmarks	32910	Specifies the document's **Bookmarks** pane.
Thumbnails	32912	Specifies the document's pages **Thumbnails** pane.
Layers	33006	Specifies the document's **Layers** pane.
Fields	33008	Specifies the document's **Fields** pane.
Comments	33204	Specifies the document's **Comments** list pane.

Remarks

Examples for usage (in pseudocode):

1. get **Thumbnails** pane visibility:

```
...
GetDocumentProperty(4095, "View.Panes["Thumbnails"].Visible",
                    DataOut, 0);
// or:
GetDocumentProperty(4095, "View.Panes[#32912].Visible",
                    DataOut, 0);
...
```

2. show **Thumbnails** pane:

```
...
```

```
SetDocumentProperty(4095, "View.Panes["Thumbnails"].Visible",
                    DataIn(1), 0);
// or:
SetDocumentProperty(4095, "View.Panes[##32912].Visible",
                    DataIn(1), 0);
...
```

See Also

Objects::Documents::<Item>.View

2.2.3.17 Document Save Methods

Supported Values

String Value	Integer Value	Description
FullOrIncrementalForDigSign	0	Defines the full saving for regular documents or incremental saving for digitally signed documents (recommended).
Full	1	Specifies the full saving for all documents always. Note: digital signature will be lost when this method is used.
Incremental	2	Specifies the incremental saving for all documents always.

See Also

Objects::Documents::SaveMethod

2.2.3.18 Document Save Modes

Supported Values

String Value	Integer Value	Description
SaveToSource	0	Document is (being) saved to source file.
SaveAs	1	Document is (being) saved to another file, control will reopen the source document to newly saved document.
SaveCopyAs	2	Document is (being) saved to another file.
SaveToStream	3	Document is (being) saved to stream.
SaveToSourceStream	4	Document is (being) saved to source stream. See also: Objects::Documents::UseStreamsDirectly.

See Also

Objects::Notifications::BeforeSaveDoc, Objects::Notifications::DocSaved

2.2.3.19 Document SaveAs Destination Types

Supported Values

String Value	Integer Value	Description

LastUsed	0	Specify it to use the last target directory in the "Save As" dialog.
Original	1	Specify it to use the current directory of the opened document in the "Save As" dialog.
Custom	2	Specify it to use the custom directory in the "Save As" dialog.

See Also

Objects::Documents::SaveAsDestType

2.2.3.20 Duplex Printing

Supported Values

String Value	Integer Value	Description
Default	-1	Default printer duplex settings.
None	0	Normal (nonduplex) printing.
Vertical	1	Long-edge binding, that is, the long edge of the page is vertical.
Horizontal	2	Short-edge binding, that is, the long edge of the page is horizontal.

See Also

Objects::Print

2.2.3.21 Export Modes

Supported Values

String Value	Integer Value	Description
ToImage	0	Defines the export of document to images with the parameters defined earlier by Objects::Export.Image.

See Also

Objects::Export.Image

2.2.3.22 Export to Image Modes

Supported Values

String Value	Integer Value	Description
AllToOneMultiImage	0	Exports all pages to one multi-page image file.
EachRangeToOneMultiImage	1	Exports each page *range* to one multi-page image file.
EachToOneImage	2	Exports each page to a separate image file.

See Also

Objects::Export.Image,

Objects::Export.Image.Mode

2.2.3.23 File Attachment Icon Types

Supported Values

String Value	Integer Value
Default	3
Attachment	
Graph	1
PaperClip	2
Tag	4

See Also

Objects::Commenting

2.2.3.24 Highlight Form Fields Masks

Supported Values

String Value	Integer Value
All	0xFFFFFFFF
None	0x00000000
Default	0x000000FC
ComboBoxes	0x00000020
CheckBoxes	0x00000008
ListBoxes	0x00000040
TextBoxes	0x00000010
PushButtons	0x00000002
RadioButtons	0x00000004
Signatures	0x00000080

See Also

Objects::Forms.HighlightFields,
Objects::Documents::<Item>.Form.HighlightFields

2.2.3.25 Image Align Types

Image Scale Types**Supported Values**

String Value	Integer Value	Description
LeftTop	0	Specifies to fit proportionally of image in the cell and align it to top left corner of the cell.
Center	1	Specifies to fit proportionally of image and center it in the cell.
RightBottom	2	Specifies to fit proportionally of image in the cell and align it to bottom right corner of the cell.
Fit	3	Specifies to full fit of image in the cell (unproportional fit).

See Also

Objects::Operations.NewDocument.FromImages.Layout

2.2.3.26 Image Conversion Types

Supported Values

String Value	Integer Value	Description
None	0	No conversion.
Grayscale	1	To Grayscale.
Monochrome	2	To Monochrome
BlackWhite	3	To Black & White

See Also

Objects::Operations.NewDocument.FromImages.Graphics

2.2.3.27 Image Scale Types

Supported Values

String Value	Integer Value	Description
None	0	No scale.
Linear	1	Linear scale.
Bilinear	2	Bilinear scale.
Bicubic	3	Bicubic scale.

See Also

Objects::Operations.NewDocument.FromImages.Layout

2.2.3.28 Image Types

Supported Image Types

String Value	Integer Value	Description
BMP*	1112363040	Windows Bitmap File Format
GIF*	1195984416	Compuserve GIF
PNG*	1347307296	Portable Network Graphic
JNG*	1246644000	JPEG Network Graphic
JPEG*	1246774599	Joint Photographic Experts Group
ICO	1229147936	Icon Image File Format
PBM*	1346522400	Portable Bitmap File Format
PGM*	1346850080	Portable Graymap
PPM*	1347439904	Portable Pixelmap
JBIG	1245858119	Joint Bi-level Image experts Group
JBIG2*	1245857586	Joint Bi-level Image experts Group v2

JPEG2K*	1246769739	JPEG 2000
JPEG2000*	1246769739	JPEG 2000
WBMP*	1463962960	Wireless Mono Bitmap File Format
PCX*	1346590752	PC Paintbrush File Format
DCX*	1145264160	Multipage PCX
TGA*	1413955872	Truevision Targa
TIFF*	1414088262	Tag Image File Format
DNG	1145980704	Digital Negative File Format
WMF	1464682016	Standard Windows Metafile
AMF	1095583264	Placeable Aldus Metafile
EMF	1162692128	Enhanced Windows Metafile

* - items represent formats which are available for read/write, unmarked items represent formats available for read only.

See Also

Objects::Export.Image,
Objects::Export.Image.Type

2.2.3.29 Keyboard Notifications Filter Flags

Supported Values

String Value	Integer Value
None	0
All	-1
Down	1
Up	2

See Also

Objects::Notifications.Keyboard

2.2.3.30 Line Ending

Supported Values

String Value	Integer Value	Description
None	0	No line ending.
Square	1	A square filled with the annotations fill color.
Circle	2	A circle filled with the annotations fill color.
Diamond	3	A diamond shape filled with the annotations fill color.
OpenArrow	4	Two short lines meeting in an acute angle to form an open arrowhead.
ClosedArrow	5	Two short lines meeting in an acute angle as in the OpenArrow style and connected by a third line to form a triangular closed arrowhead filled with the annotations fill color.
Butt	6	A short line at the endpoint perpendicular to the line itself.

ROpenArrow	7	Two short lines in the reverse direction from OpenArrow.
RClosedArrow	8	A triangular closed arrowhead in the reverse direction from ClosedArrow.
Slash	9	A short line at the endpoint approximately 30 degrees clockwise from perpendicular to the line itself.

See Also

Objects::Commenting

2.2.3.31 Link Highlight Mode

Supported Values

String Value	Integer Value	Description
None	0	No highlighting.
Invert	1	Invert the contents of the link rectangle.
Outline	2	Invert the link's border.
Push	3	Displays the link's down appearance.

See Also

Objects::Commenting

2.2.3.32 Main Bars

Supported Main Bars Identifiers

Name	ID	Description
Menu	33009	Specifies the main **Menu** bar.
Tab	32940	Specifies the documents navigation **Tab** bar.
Status	59393	Specifies the main **Status** bar.
File	32908	Specifies the **File** toolbar.
RotateView	32929	Specifies the **Rotate View** toolbar.
Standard	32924	Specifies the **Standard** toolbar.
Zoom	32925	Specifies the **Zoom** (pages magnification) toolbar.
Find	33000	Specifies the **Find** toolbar.
CommentAndMarkup	33138	Specifies the **Comment And Markup** toolbar.
Promoting	32913	Specifies the **Promotion** bar.
Properties	33225	Specifies the **Properties** toolbar.
Links	36342	Specifies the **Links** toolbar.
Measuring	36345	Specifies the **Measuring** toolbar.

Remarks

Examples for usage (in pseudocode):

1. get **File** toolbar visibility:

```
...
GetProperty("View.Bars["File"].Visible", DataOut, 0);
```

```
// or:
GetProperty("View.Bars[#32908].Visible", DataOut, 0);
...
```

2. show **File** toolbar:

```
...
SetProperty("View.Bars["File"].Visible", DataIn(1), 0);
// or:
SetProperty("View.Bars[#32982].Visible", DataIn(1), 0);
...
```

See Also

Objects::View

2.2.3.33 Main Panes

Supported Main Panes Identifiers

Name	ID	Description
Search	32982	Specifies the full **Search** pane.

Remarks

Examples for usage (in pseudocode):

1. get **Search** pane visibility:

```
...
GetProperty("View.Panes["Search"].Visible", DataOut, 0);
// or:
GetProperty("View.Panes[#32982].Visible", DataOut, 0);
...
```

2. show **Search** pane:

```
...
SetProperty("View.Panes["Search"].Visible", DataIn(1), 0);
// or:
SetProperty("View.Panes[#32982].Visible", DataIn(1), 0);
...
```

See Also

Objects::View

2.2.3.34 Mouse Notifications Filter Flags

Supported Values

String Value	Integer Value
None	0
All	-1
Move	1
Down	2
Up	4
Wheel	8

See Also

Objects::Notifications.Mouse

2.2.3.35 Name Generation Macros

Supported Macros

Name	Description
Document Title	Specifies UI title of the current document.
Page Number	Specifies index (1-based) of the current page.
PAGE	The same as *Page Number* macro.
PAGES	Specifies pages number of the current document.
File Name	Specifies file title of the current document.
Date	Specifies current date in format MM-DD-YYYY.
Year	Specifies current year.
Month	Specifies current month.
Day	Specifies current day of month.
Time	Specifies current time in format HH-MM-SS.
Hour	Specifies current hour.
Minute	Specifies current minute.
Second	Specifies current second.
Auto Number	Specifies inserting of a special numerical string to guarantee a unique result-name.

Remarks

In string expression each macros should be bracketed by "<" and ">" symbols.
Some examples:

```
...
"<Document Title>_<PAGE>_of_<PAGES>",
"<File Name> <Date> <Auto Number>",
"MyFile Created: Date is [<Date>], Time is [<Time>]",
...
```

For example (in pseudocode), set special macros list to generate unique image files during export process:

```
...
// for document "C:\Test.pdf":
SetProperty("Export.Image.FileName",
            DataIn("<File Name>_<Date>_<Auto Number>"), 0);
// newly created files, for example:
// C:\Test_09-17-2007_01.jpg
// C:\Test_09-17-2007_02.jpg
// C:\Test_09-17-2007_03.jpg
...
```

See Also

Objects::Export.Image

2.2.3.36 New Paragraph Modes

Supported Values

String Value	Integer Value	Description
Auto	0	Auto-detect paragraphs.
One2NP	1	Each newline character starts a new paragraph.

Double2NP	2	Double newline character starts a new paragraph, singles ignored.
Double2NPOne2Space	3	Double newline character starts a new paragraph, single converted to space.

See Also

Objects::Operations.NewDocument.FromText

2.2.3.37 Pages Layouts

Supported Values

String Value	Integer Value	Description
Automatic	-1	Specifies the selection of page layout automatically.
Auto		
Continuous	1	Specifies the displaying of one page at a time in continuous mode (when scrolling down shows the beginning of the next page under the end of the current).
Continuous-Facing	7	Specifies the displaying of pages by two in continuous mode.
ContinuousFacing		
Facing	6	Specifies the displaying of two pages at a time.
SinglePage	0	Specifies the displaying of just one page at a time.
Single		

See Also

Objects::PageDisplay.DefaultLayout,
Objects::Documents::<Item>.Pages.Layout

2.2.3.38 Pages Magnification Modes

Supported Values

String Value	Integer Value	Description
Percent	0	Set zoom level in percents.
ActualSize	1	Displays pages in actual size (100%).
FitWidth	2	Specifies fitting page width in the window.
FitPage	3	Fits total page content in the window.
FitVisible	4	Not yet available.

See Also

Objects::Documents::<Item>.Pages.ZoomMode

2.2.3.39 Pages Magnifications

Supported Values

String Value	Integer Value	Description

Automatic Auto	0	Selects page zoom automatically.
ActualSize	100	Displays pages in actual size (100%).
FitPage	-1	Fits total page content in the window.
FitWidth	-2	Specifies fitting page width in the window.
FitVisible	-3	Not yet available.

Remarks

These values are basic only. You can use custom values also.
For example, you can specify the string values: "99**.**9**%**", "99**,**9" or corresponding numeric values.

See Also

Objects::PageDisplay.DefaultZoom,
Objects::Documents::<Item>.Pages.Zoom

2.2.3.40 Paper Modes

Supported Values

String Value	Integer Value	Description
Custom	0	Use custom paper sizes.
Standard	1	Use standard paper sizes.

See Also

Objects::Operations.NewDocument.FromBlank
Objects::Operations.InsertEmptyPages

2.2.3.41 Paper Orientation

Supported Values

String Value	Integer Value	Description
Default	0	Default printer's paper-orientation settings.
Portrait	-1	Normal paper orientation (portrait).
Landscape	-2	Landscape paper orientation.

See Also

Objects::Print

2.2.3.42 PDF-Specification Versions

Supported Values

String Value	Integer Value
Auto	0
1.4	0x14

1.5	0x15
1.6	0x16
1.7	0x17

See Also

Objects::Operations.NewDocument

2.2.3.43 Print Notifications Filter Flags

Supported Values

String Value	Integer Value
None	0
All	-1
BeginDocument	1
EndDocument	2
BeginSheet	4
EndSheet	8
BeginPage	16
EndPage	32
Prepare	64

Remarks

The "Prepare" notify called once before start of printing.
At this stage you also can change "Notifications.Print.Copies" and "Notifications.Print.Collate" properties.

See Also

Objects::Notifications.Print

2.2.3.44 Print Scale Types

Supported Values

String Value	Integer Value	Description
None	0	No page scaling. Print as original. When printing a page may not fit in a chosen paper.
No		
FitToMargins	1	If a page appears to be too big for the given paper it will be reduced. If a page appears to be too small for the given paper it will be increased. In both cases, printer margins for chosen paper are taken into account.
ReduceToMargins	2	The size of a page will be reduced if the page is too big for the given paper. If the page fits the paper its size will not be changed. The printer margins for chosen paper are taken into account.
TileLarge	4	Allows to print PDF page onto multiple sheets of paper if the

		page is larger than the page sizes available on your printer.
TileAll	5	Allows to print all PDF pages onto multiple sheets of paper.
Mult Multiple	6	Multiple pages can be printed on the same sheet of paper.
Booklet Book	7	Allows to create a booklet with pages arranged on sheets of paper so that they are in the correct order when the paper is collated, folded, and stapled.

See Also

Objects::Print.ScaleType

2.2.3.45 Print Specials

Supported Values

String Value	Integer Value	Description
Default DocumentAndMarkups	33550369 (0x01FFF021)	Print document content and all markups.
Document	33 (0x00000021)	Print document content only.
DocumentAndStamps	4194337 (0x00400021)	Print document content and stamps.
FieldsDataOnly	2	Print field data only.

See Also

Objects::Print.PrintSpec

2.2.3.46 Print Text as Curves

Supported Values

String Value	Integer Value	Description
Auto	0	With this option PDF-XChange Viewer decides what text should be printed as curves.
ForEmbeddedFonts	1	Only embedded fonts will be printed as curves.
Always	2	Text will always be printed as curves.

See Also

Objects::Print.TextAsCurves

2.2.3.47 Range Filters

Supported Values

String Value	Integer Value	Description
All	1	All items.

Odd	4	Odd items.
Even	5	Even items.

* - for more operations the "items" can represent pages of document (Print, Export, etc.).

See Also

Objects::Print.RangeFilter,
Objects::Export.Image.RangeFilter,
Objects::Operations::SummarizeAnnots

2.2.3.48 Range Types

Supported Values

String Value	Integer Value	Description
None	0	No items*.
All	1	All items.
First	2	First item.
Last	3	Last item.
Odd	4	Odd items.
Even	5	Even items.
Exact	6	*Items* specified by the *range specification* in **RangeText**, for example **"1, 3, 5-7, 9"**.
Current	7	Use current item.
Selected	9	Use selected items.

* - for more operations the "*items*" can represent pages of document (Print, Export, etc.).

See Also

Objects::Print.RangeType,
Objects::Export.Image.RangeType,
Objects::Operations::SummarizeAnnots

2.2.3.49 Registry Roots

Supported Values

String Value	Description
HKCR	Defines the **HKEY_CLASSES_ROOT** registry root.
HKEY_CLASSES_ROOT	
HKCU	Defines the **HKEY_CURRENT_USER** registry root.
HKEY_CURRENT_USER	
HKLM	Defines the **HKEY_LOCAL_MACHINE** registry root.
HKEY_LOCAL_MACHINE	

See Also

Operations::LoadSettings,
Operations::SaveSettings

2.2.3.50 Rotation Direction

Supported Values

String Value	Integer Value	Description
Clockwise	0	Clockwise 90 degrees
CounterClockwise	1	Counterclockwise 90 degrees
180	2	180 degrees

See Also

Objects::Operations.RotatePages

2.2.3.51 Search Modes

Supported Values

String Value	Integer Value	Description
InActive	0	Search in active PDF document.
InOpened	1	Search in all opened documents.
InFolder	2	Search in folder.

See Also

Objects::Search.Where

2.2.3.52 Selection Notifications Filter Flags

Supported Values

String Value	Integer Value
None	0
All	-1
Annotations	1
Text	2

See Also

Objects::Notifications.Selection

2.2.3.53 Shortcut Key Types

Supported Values

String Value	Integer Value
Virt	1
ASCII	0

See Also

Objects::Commands::<Item>.Shortcut

2.2.3.54 Shortcut Modifiers

Supported Values

Integer Value	Description
16	The **ALT** key must be held down when the accelerator key is pressed.
8	The **CTRL** key must be held down when the accelerator key is pressed.
4	The **SHIFT** key must be held down when the accelerator key is pressed.
128	The flag for default shortcuts. If it present then shortcut is default.

See Also

Objects::Commands::<Item>.Shortcut

2.2.3.55 Sticky Note Icon Types

Supported Values

String Value	Integer Value
Default	3
Comment	
Checkmark	1
Circle	2
Cross	4
Help	5
Insert	6
Key	7
NewParagraph	8
TextNote	9
Paragraph	10
RightArrow	11
RightPointer	12
Star	13
UpArrow	14
UpLeftArrow	15

See Also

Objects::Commenting

2.2.3.56 Summarize Annotations Group Types

Supported Values

String Value	Integer Value

Page	0
Type	1
Author	2
Date	3
Subject	4

See Also

Objects::Operations.SummarizeAnnots

2.2.3.57 Summarize Annotations Output Types

Supported Values

String Value	Integer Value
PDF	0
RTF	1
TXT	2
HTML	3

See Also

Objects::Operations.SummarizeAnnots.Output

2.2.3.58 Summarize Annotations PDF Layouts

Supported Values

String Value	Integer Value	Description
CommentsOnly	0	Comments only.
DocAndCommentsWithConnectors	3	Document and comments with connector lines on single page.
DocAndCommentsWithSeqNumbers	5	Document and comments with sequence numbers on single page.
DocAndComments	1	Document and comments on single page.
DocAndCommentsWithConnectorsOnTwoPages	11	Document and comments with connector lines on separate pages.
DocAndCommentsWithSeqNumbersOnTwoPages	13	Document and comments with sequence numbers on separate pages.
DocAndCommentsOnTwoPages	9	Documents and comments on separate pages.

See Also

Objects::Operations.SummarizeAnnots.Output.PDF

2.2.3.59 Text Align

Supported Values

String Value	Integer Value	Description
Left	0	Left text align.
Center	1	Center text align.
Right	2	Right text align.
Justify	3	Justify text align.

See Also

Objects::Text Format

2.2.3.60 Text Editor Notifications Filter Flags

Supported Values

String Value	Integer Value
None	0
All	-1
Begin	1
End	2

See Also

Objects::Notifications.Print

2.2.3.61 Text File Placing Modes

Supported Values

String Value	Integer Value	Description
NewPara	0	Start each file from new paragraph.
NewPage	1	Start each file from new page.

See Also

Objects::Operations.NewDocument.FromText

2.2.3.62 Text Rendering Mode

Supported Values

String Value	Integer Value	Description
Fill	0	Fill text.
Stroke	1	Stroke text.
FillStroke	2	Fill, then stroke text.

See Also

Objects::Text Format

2.2.3.63 Tools

Supported Tools Identifiers

Name	ID
Hand	32613
Select	32620
Snapshot	32614
ZoomIn	32610
ZoomOut	32611
Loupe	32609
StickyNote	33132
TypeWriter	33226
Callout	33201
TextBox	33202
Highlight	33221
StrikeOut	33223
Underline	33222
Arrow	33123
Line	33124
Rect	33125
Oval	33126
PolyLine	33128
Polygon	33127
Cloud	33130
Pencil	33129
Eraser	36300
Stamp	33131
Distance	36346
Perimeter	36347
Area	36348
Link	36343
QuadLink	36344

See Also

Objects::Tools

2.2.3.64 User Choices

Supported Values

String Value	Integer Value	Description
Default	0	Continue default processing.
OK	1	-
Okay	1	-

Cancel	2	-
Abort	3	-
Retry	4	-
Ignore	5	-
Yes	6	-
No	7	-
YesToAll	100	-
NoToAll	101	-
SaveAs	103	-
Auto	102	Special code that instructs generation of unique names to automatically output files. See Objects::Prompts.ConfirmFileReplace.
Automatic	102	The same as *Auto*.

See Also

Objects::Prompts

2.2.3.65 User Interface Languages

Supported Values

String Value	Integer Value	Description
Auto System	0	Selects the user's regional settings in UI.
Default	-1	Selects the application's default language, see also the Objects::International.DefaultLocaleID.
BuiltIn	-2	Selects the application's built-in language (English(US)).
English(US) English	0x0409	Selects the English(US) language in UI.
German(DE) German	0x0407	Selects the German(DE) language in UI.
French(FR) French	0x040C	Selects the French(FR) language in UI.
Russian	0x0419	Selects the Russian language in UI.
Ukrainian	0x0422	Selects the Ukrainian language in UI.

Remarks

These values are basic only.
For other supported languages please look at **"<FolderOfExeModule>\Languages\"** folder.
See also Objects::General.ApplicationModulePath.

See Also

Objects::International.LocaleID

2.2.4 Simple Notifications

This section represents a groups of special named notifications:

- Global
- Text Editor
- Select Tool

Remarks

Example for usage(in pseudocode):

```
// to prevent keyboard input:
...
function OnEvent(Type, Name, DataIn, DataOut, Flags)
{
   if (Type == PXCVA_OnNamedNotify) and (Name == "Global::CheckKey") then
   {
      ...
      DataOut = 0;
   }
}
...
```

2.2.4.1 Global

Supported Global Named Notifications

Name	Description
Global::OperationsHistoryChanged	Signals if operations history has been changed (Undo/ Redo).
Global::CheckSysKey	Control sends this notification to the client when the user holds down the ALT key and then presses another key (WM_SYSKEYDOWN windows message). The **DataIn** parameter of the OnEvent procedure contains the virtual key code. To prevent the default processing - write **0** (zero) to the **DataOut** parameter.
Global::CheckKey	Control sends this notification to the client when the user presses any key (WM_KEYDOWN windows message). The **DataIn** parameter of the OnEvent procedure contains the virtual key code. To prevent the default processing - write **0** (zero) to the **DataOut** parameter.
Global::FocusGained	This simple event informs you: the control (any window from control) gained or already have the input focus. This event is periodically sent to the client when the end-user interacts with the control - clicks by mouse, activates any window of control, etc.

2.2.4.2 Select Tool

Supported Named Notifications

Name	Description
SelectTool::OnSelectionChanged	Signals if selection in active document has been changed.

Remarks

This notification is **obsolete**. Please look to Objects::Notifications.Selection.

2.2.4.3 Text Editor

Supported Named Notifications

Name	Description
TextEditor:: OnStart	Signals if *Text Editor* has just started.
TextEditor::OnStop	Signals if *Text Editor* has just stopped.

Remarks

These notifications are **obsolete**. Please look to Objects::Notifications.TextEditor.

2.3 JavaScript Support

Currently supported JavaScript objects, methods and properties are listed below, sorted by object name in alphabetic order.

For more information, please refer to the JavaScript for Acrobat API Reference.

Supported Objects

Name	Description
Annotation	This object represents a PDF annotation.
app	A static JavaScript object that represents the PDF-XChange Viewer control. It defines a number of viewer-specific functions plus a variety of utility routines and convenience functions.
Bookmark	This object represents a node in the bookmark tree that appears in the bookmarks navigational panel (see also Document Panes). Bookmarks are typically used as a table of contents allowing the user to navigate quickly to topics of interest.
console	This object is a static object that enables an access to the JavaScript console for executing JavaScript and displaying debug messages.
Doc	This object provides the interface between a PDF document open in the viewer and the JavaScript interpreter. It provides methods and properties for accessing PDF document.
event	All JavaScript scripts are executed as the result of a particular *event*. For each of these events, JavaScript creates an event object. During the occurrence of each event, you can access this object to get and possibly manipulate information about the current state of the event.
Field	This object represents a PDF form field. In the same manner that a form author can modify an existing field's properties, such as border color or font, the JavaScript user can use the **Field** object to perform the same modifications.
FullScreen	The interface to *fullscreen* (presentation mode) preferences and properties.
global	This is a static JavaScript object that allows you to share data between documents and to have data be persistent across sessions. Such data is called *persistent global data*. Global data-sharing and notification across documents is done through a subscription mechanism, which allows you to monitor global data variables and report their value changes across documents.

identity	This is a static object that identifies the current user of the application.
Link	This object is used to set and get the properties and to set the JavaScript action of a link.
OCG	An OCG object represents an *optional content group* in a PDF file. Content in the file can belong to one or more optional content groups. Content belonging to one or more OCGs is referred to as optional content and its visibility is determined by the states (ON or OFF) of the OCGs to which it belongs. In the simplest case, optional content belongs to a single OCG, with the content being visible when the OCG is on and hidden when the OCG is off. More advanced visibility behavior can be achieved by using multiple OCGs and different visibility mappings.
Template	Template objects are named pages within the document. These pages may be hidden or visible and can be copied or spawned. They are typically used to dynamically create content (for example, to add pages to an invoice on overflow).
util	A static JavaScript object that defines a number of utility methods and convenience functions for string and date formatting and parsing.

See Also

IPDFXCview::RunJavaScript

2.3.1 Annotation

This object represents an PDF-annotation.

Supported Properties

```
All are supported.
```

Supported Methods

```
destroy
getProps
setProps
```

See Also

IPDFXCview::RunJavaScript,
JavaScript for Acrobat API Reference

2.3.2 app

A static JavaScript object that represents the PDF-XChange Viewer control.

It defines a number of document-specific functions plus a variety of utility routines and convenience functions.

Supported Properties

```
activeDocs
calculate
```

```
formsVersion
fs
fullscreen
language
numPlugIns
platform
plugIns
printColorProfiles
printerNames
thermometer
toolbar
toolbarHorizontal
toolbarVertical
viewerType
viewerVariation
viewerVersion
```

Supported Methods

```
alert
beep
execMenuItem
getPath
mailMsg
newDoc
openDoc
popUpMenu
popUpMenuEx
responce
clearInterval
clearTimeOut
setIterval
setTimeOut
```

See Also

IPDFXCview::RunJavaScript,
JavaScript for Acrobat API Reference

2.3.3 Bookmark

This object represents a node in the bookmark tree that appears in the bookmarks navigational panel (see also Document Panes).

Bookmarks are typically used as a table of contents allowing the user to navigate quickly to topics of interest.

Supported Properties

```
All are supported.
```

Supported Methods

```
All are supported.
```

See Also

IPDFXCview::RunJavaScript,
JavaScript for Acrobat API Reference

2.3.4 console

This object is a static object that enables access to the JavaScript console for executing JavaScript and displaying debug messages.

Supported Properties

```
There are no properties for
this object.
```

Supported Methods

```
All are supported.
```

See Also

IPDFXCview::RunJavaScript,
JavaScript for Acrobat API Reference

2.3.5 Doc

This object provides the interface between a PDF document open in the viewer and the JavaScript interpreter.
It provides methods and properties for accessing the PDF document.

Supported Properties

```
author
baseURL
calculate
creationDate
creator
dataObjects
dirty
docID
documentFileName
external
filesize
innerAppWindowRect
innerDocWindowRect
keywords
```

layout
modDate
mouseX
mouseY
numFields
numPages
outerAppWindowRect
outerDocWindowRect
pageNum
path
pageWindowRect
permStatusReady
producer
securityHandler
selectedAnnots
subject
templates
title
URL
viewState
xfa
XFAForeground
zoom
zoomType

Supported Methods

addAnnot
addLink
calculateNow
closeDoc
deletePages
exportAsFDF
exportAsFDFStr
exportAsXFDF
exportAsXFDFSt
exportDataObject
extractPages
flattenPages
getAnnots

getAnnots3D
getField
getLinks
getNthFieldName
getOCGs
getPageBox
getPageNthWord
getPageNthWordQuads
getPageNumWords
getPageRotation
getPrintParams
gotoNamedDest
flattenPages
importAnFDF
importAnXFDF
importDataObject
insertPages
movePage
newPage
print
removeDataObject
removeLinks
resetForm
scroll
selectPageNthWord
setAction
setPageRotations
spawnPageFromTemplate
submitForm
syncAnnotScan

See Also

IPDFXCview::RunJavaScript,
JavaScript for Acrobat API Reference

2.3.6 event

All JavaScript scripts are executed as the result of a particular event.

For each of these events, JavaScript creates an event object. During the occurrence of each event, you can access this object to get and possibly manipulate information about the current state of the event.

Supported Properties

```
All are supported.
```

Supported Methods

```
All are supported.
```

See Also

IPDFXCview::RunJavaScript,
JavaScript for Acrobat API Reference

2.3.7 Field

This object represents a PDF-form field.

In the same manner that a form author can modify an existing field's properties, such as border color or font, the JavaScript user can use the **Field** object to perform the same modifications.

Supported Properties

```
All are supported.
```

Supported Methods

```
browseForFileToSubmit
buttonGetCaption
buttonSetCaption
clearItems
getArray
getItemAt
insertItemAt
deleteItemAt
isBoxChecked
isDefaultChecked
checkThisBox
defaultIsChecked
setItems
setAction
setFocus
```

See Also

IPDFXCview::RunJavaScript,
JavaScript for Acrobat API Reference

2.3.8 FullScreen

The interface to *fullscreen* (presentation mode) preferences and properties.

Supported Properties

```
All are supported.
```

Supported Methods

```
There are no methods for this
object.
```

See Also

IPDFXCview::RunJavaScript,
JavaScript for Acrobat API Reference

2.3.9 global

This is a static JavaScript object that allows you to share data between documents and to have data be persistent across sessions.

Such data is called *persistent global data*. Global data-sharing and notification across documents is done through a subscription mechanism, which allows you to monitor global data variables and report their value changes across documents.

Supported Properties

```
All are supported.
```

Supported Methods

```
All are supported.
```

See Also

IPDFXCview::RunJavaScript,
JavaScript for Acrobat API Reference

2.3.10 identity

This is a static object that identifies the current user of the application.

Supported Properties

```
All are supported.
```

Supported Methods

```
There are no methods for this
object.
```

See Also

IPDFXCview::RunJavaScript,
JavaScript for Acrobat API Reference

2.3.11 Link

This object is used to set and get the properties and to set the JavaScript action of a link.

Supported Properties

```
All are supported.
```

Supported Methods

```
All are supported.
```

See Also

IPDFXCview::RunJavaScript,
JavaScript for Acrobat API Reference

2.3.12 OCG

An OCG object represents an *optional content group* in a PDF file.

Content in the file can belong to one or more optional content groups. Content belonging to one or more OCGs is referred to as optional content and its visibility is determined by the states (ON or OFF) of the OCGs to which it belongs. In the simplest case, optional content belongs to a single OCG, with the content being visible when the OCG is on and hidden when the OCG is off. More advanced visibility behavior can be achieved by using multiple OCGs and different visibility mappings.

Supported Properties

```
All are supported.
```

Supported Methods

```
All are supported.
```

See Also

IPDFXCview::RunJavaScript,
JavaScript for Acrobat API Reference

2.3.13 Template

Template objects are named pages within the document.

These pages may be hidden or visible and can be copied or spawned. They are typically used to dynamically create content (for example, to add pages to an invoice on overflow).

Supported Properties

```
All are supported.
```

Supported Methods

```
All are supported.
```

See Also

IPDFXCview::RunJavaScript,
JavaScript for Acrobat API Reference

2.3.14 util

A static JavaScript object that defines a number of utility methods and convenience functions for string and date formatting and parsing.

Supported Properties

```
There are no properties for
this object.
```

Supported Methods

```
crackURL
printd
printf
printx
spansToXML
scand
xmlToSpans
```

See Also

IPDFXCview::RunJavaScript,
JavaScript for Acrobat API Reference

2.4 Enumerations

Enumerations

Enumeration	Description
PXCVA_DocumentSaveFlags	The flags shown in this enumeration allow you to use special document save modes (see also IPDFXCview::SaveDocument).
PXCVA_EventType	The values shown in this enumeration allow you to identify different types of received events (see also _IPDFXCviewEvents::OnEvent).
PXCVA_Flags	The flags shown in this enumeration allow you to condition the behaviour of the various actions. Combinations of these flags are usually passed in the last **Flags** argument.
PXCVA_ViewObjectType	The values shown in this enumeration allow you to determine different types of view objects (see also IPDFXCview:: GetViewObjectFromName).

See Also

IPDFXCview::DoVerb,
IPDFXCview::SaveDocument,
_IPDFXCviewEvents::OnEvent

2.4.1 PXCVA_DocumentSaveFlags

The flags shown in this enumeration allow you to use special document save modes.

See also IPDFXCview::SaveDocument.

Syntax

```
enum PXCVA_DocumentSaveFlags
{
```

```
        PXCVA_DocumentSaveAs        = 0x00000001,
        PXCVA_DocumentSaveCopyAs    = 0x00000003,
        PXCVA_DocumentSaveInc       = 0x00000004,
        PXCVA_DocumentSaveForWeb    = 0x00000008
};
```

Constants

PXCVA_DocumentSaveAs
: The specified document may be saved with a new name and/or to a new location. If the operation succeeds then the original document will be closed and replaced by newly saved document in the Viewer.

PXCVA_DocumentSaveCopyAs
: The specified document will be saved to a new location as new copy. The default name for the saved file is **<DocumentName> copy[(<Index>)].pdf**.
For example:
if original file is c:\test.pdf, then saved copy can be: c:\test copy.pdf, c:\test copy(2).pdf, c:\test copy(3).pdf, ...

PXCVA_DocumentSaveInc
: The original content will be written without any changes first, and then *all new or changed objects will be appended*. N.B. This flag MUST be used to save all digitally signed documents, or the digital signature will be lost.

PXCVA_DocumentSaveForWeb
: The specified document will be saved in a special format that is adapted for fast Web-view. Not implemented.

See Also

IPDFXCview::SaveDocument

2.4.2 PXCVA_EventTypes

The values shown in this enumeration allow you to identify different types of received events.

See also _IPDFXCviewEvents::OnEvent.

Syntax

```
enum PXCVA_EventTypes
{
        PXCVA_OnPropertyChanged     = 0x00000001,
        PXCVA_OnDisplayPrompt       = 0x00000003,
        PXCVA_OnNamedNotify         = 0x00000004,
};
```

Constants

PXCVA_OnPropertyChanged
: Signalled if the property just changed. to obtain actual value of the property call IPDFXCview::GetProperty method.

PXCVA_OnDisplayPrompt
: Used for confirmation and customize of UI prompt (dialog) before display. For more information, see Objects::Prompts.

PXCVA_OnNamedNotify
: Used for notification about more common/special events.
For more information, see Objects::Notifications.

See Also

_IPDFXCviewEvents::OnEvent

2.4.3 PXCVA_Flags

The flags shown in this enumeration allow you to condition the behaviour of the various actions. Combinations of these flags are usually passed in the last **Flags** argument.

Syntax

```
enum PXCVA_Flags
{
	PXCVA_NoApply		= 0x00000001,
	PXCVA_NoUI		= 0x00000002,
	PXCVA_GetNamed		= 0x00000004,
	PXCVA_Sync		= 0x00000008,
	PXCVA_OutArgs		= 0x00000010,
};
```

Constants

PXCVA_NoApply
: The changes should not be applied immediately. Changes are to be accumulated before any action will run or the IPDFXCview::ApplyAllCachedChanges method call is used. See Operations::SetNoApply also.

PXCVA_NoUI
: This flag disables the UI. For example, when opening a corrupted document the ActiveX control will normally display a dialog to announce this fact. When this flag is set then no messages will be displayed. In this case the error could be detected by checking the return result of the function.

PXCVA_GetNamed
: When specified, the property *named value (simple string)* will be returned in **DataOut**. See Operations::GetNamed also.

PXCVA_Sync
: In time of execution of some methods the control automatically simulates synchronous call to partially unlock the client during the long-term call. For example you start document printing. By default the control shows the print-dialogue and co-operates with the user for some period of time, after what it prints. In this case the call of method **PrintDocument** will be executed for the indefinitely long period of time and if this method is really synchronous the client will be completely locked for the period of interaction among the user and the printing. That is the user interface of the client will not be drawn. To avoid it, the control by default simulates synchronous call of such method by internal messages loop. During this interactivity will be locked for the client but the drawing of client's user interface will be unlocked.
To disable automatic simulation of synchronous calls, specify this flag.

PXCVA_OutArgs
: When specified, the pointer to a ***new*** object with IPDFXCargs interface will be returned in **DataOut**. Use this flag for simplify reading of a data array(**SAFEARRAY**) from the **DataOut** .

See Also

IPDFXCview::DoVerb, IPDFXCview::GetProperty, IPDFXCview::SetProperty, IPDFXCview::ApplyAllCachedChanges, IPDFXCview::DiscardAllCachedChanges

2.4.4 PXCVA_ViewObjectTypes

The values shown in this enumeration allow you to determine different types of view objects.

See also IPDFXCview::GetViewObjectFromName.

Syntax

```
enum PXCVA_ViewObjectTypes
{
	PXCVA_Bar	= 0x00000000,
	PXCVA_Pane	= 0x00000001,
};
```

Constants

PXCVA_Bar
Signalled if detected view object is a command bar: toolbar, menu bar, status bar.

PXCVA_Pane
Signalled if detected view object is a pane: special UI container which contains another advanced view.

See Also

_IPDFXCviewEvents::OnEvent, IPDFXCview::GetViewObjectFromName, Objects::View, Objects::Documents::<Item>.View

2.5 Simple ActiveX Control

This ActiveX control has been specially developed for fast and simple usage, does not require the developer's licence, but has the limited functionality.

Note: this control does not require the developer's licence and also works with the end-user's pro-licence which can be installed on end-user's computer.
So, if the end-user uses the pro-licence the control will work in a pro-mode on his computer, if the user has no pro-licence the control will work in free-mode on his computer.

The Simple ActiveX control supports the following interfaces:

IPDFXCpreview	This interface allows applications to implement an instance of the **Simple PDF-XChange Viewer control**.

Control Identificators:

```
CLSID: {DFC89414-E8E7-49c0-B14E-EB61245D3A5D}
ProgID: PDFXCviewAx.CoPDFXCpreview
```

2.5.1 IPDFXCpreview

This interface is the **default** interface of our **Simple ActiveX control** (Microsoft ActiveX control). This interface is derived directly from **IDispatch**.

- Methods
- Properties

Interface Identificator:

```
GUID(IID_IPDFXCpreview): {C7BCFE0A-B521-4181-A722-904DBF9427D8}
```

Requirements

OS Versions: Windows 2000 and later.
TypeLib: PDFXCviewAx.tlb

2.5.1.1 Methods

Methods

Method	Description
Print	Prints the opened document.

See Also

IPDFXCview::PrintDocument

2.5.1.1.1 Print

Prints the opened document.

Syntax

```
HRESULT Print(VARIANT_BOOL fSilent);
```

Parameters

fSilent
[in] **VARIANT_BOOL** that specifies special flag for print without displaying of UI-dialog.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCview::PrintDocument

2.5.1.2 Properties

Properties

Property Name	Description
AllowAccelerators	Allows or denies all control's accelerators.
LockedView	Allows or denies all UI-customization.
ReadOnly	Allows or denies all modification operations, for all documents.
SettingsURL	Sets or gets the *URL* to the *settings file* for control customizing.
Src	Sets/gets the *source-URL* of the document for open.

See Also

IPDFXCview2

2.5.1.2.1 AllowAccelerators

Allows or denies all control's accelerators.

Syntax

```
HRESULT get_AllowAccelerators(VARIANT_BOOL* ValueOut);
HRESULT put_AllowAccelerators(VARIANT_BOOL ValueIn);
```

Parameters

ValueOut
[out] Pointer to a **VARIANT_BOOL** that receives flag for accelerators usage.
ValueIn
[in] **BSTR** that specifies flag for accelerators usage.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

Objects::General.AllowAllAccelerators

2.5.1.2.2 LockedView

Allows or denies all UI-customization.

Syntax

```
HRESULT get_LockedView(VARIANT_BOOL* ValueOut);
HRESULT put_LockedView(VARIANT_BOOL ValueIn);
```

Parameters

ValueOut
[out] Pointer to a **VARIANT_BOOL** that receives flag for view locking.
ValueIn
[in] **BSTR** that specifies flag for view locking.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

Objects::View.Locked

2.5.1.2.3 ReadOnly

Allows or denies all modification operations, for all documents.

Syntax

```
HRESULT get_ReadOnly(VARIANT_BOOL* ValueOut);
HRESULT put_ReadOnly(VARIANT_BOOL ValueIn);
```

Parameters

ValueOut
[out] Pointer to a **VARIANT_BOOL** that receives *read-only* flag.
ValueIn

[in] **BSTR** that specifies *read-only* flag.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

Objects::General.DenyAllModifyOperations

2.5.1.2.4 SettingsURL

Sets or gets the URL to the settings.dat file with all stored settings for the control. Simply specify the correct path location to the file and the control will load and use these stored settings.

Example:

http://www.mysite.com/settings.dat
file:///C:/settings.dat
C:\settings.dat

Syntax

```
HRESULT get_SettingsURL(BSTR* ValueOut);
HRESULT put_SettingsURL(BSTR ValueIn);
```

Parameters

ValueOut
[out] Pointer to a **BSTR** that receives source-URL of the file with control's settings.
ValueIn
[in] **BSTR** that specifies source-URL of the file with control's settings.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCview::LoadSettings,
Operations::LoadSettings

2.5.1.2.5 Src

Sets or gets the *source-URL* of a document. Simply specify correct path to a document and the control will open the required file.
The source-URL may contain some specific 'open' actions(see PDF Open Parameters reference), for example:

http://www.mysite.com/test.pdf#page=2&zoom=150
file:///C:/test.pdf
C:\test.pdf

Syntax

```
HRESULT get_Src(BSTR* ValueOut);
HRESULT put_Src(BSTR ValueIn);
```

Parameters

ValueOut
[out] Pointer to a **BSTR** that receives source-URL of the document.
ValueIn
[in] **BSTR** that specifies source-URL of the document.

Return Value

Returns **S_OK** if successful, or an error value otherwise. To obtain the text description of a received error code, use GetTextFromResult.

See Also

IPDFXCview::OpenDocument,
Operations::OpenDocument

3 How To Use

This topic contains the following sections:

- How to Open a Document?
- How to Print a Document?
- How to Save a Document?
- How to Close a Document?
- How to Disable a Command?
- How to Enumerate Characters in Document?
- How to Extract Text from Document?
- How to Summarize Annotations from Document?

3.1 How to Open a Document?

To use examples below you must add the ActiveX control to your project and obtain a pointer to a IPDFXCview interface.

C++ (with ATL)

```
CComPtr<IPDFXCview> spViewer;
...
CComBSTR bsFileName(L"C:\\Test.pdf");
CComBSTR bsPassword(L"123");
LONG nDocumentID = 0;
HRESULT hRes = spViewer->OpenDocument(bsFileName.m_str,
                    bsPassword.m_str, &nDocumentID, 0);
```

C#

```
int nDocumentID;
int nRes;
try
{
       axCoPDFXCview1.OpenDocument("C:\\Test.pdf", "123", out nDocumentID, 0);
}
```

```
catch (Exception ex)
{
	nRes = System.Runtime.InteropServices.Marshal.GetHRForException(ex);
}
```

VB.NET

```
Dim nDocumentID As Integer
Dim nRes As Integer
Try
	AxCoPDFXCview1.OpenDocument("C:\Test.pdf", "123", nDocumentID, 0)
Catch ex As Exception
	nRes = System.Runtime.InteropServices.Marshal.GetHRForException(ex)
End Try
```

VB 6.0

```
Dim nDocumentID As Long
Call CoPDFXCview1.OpenDocument("C:\Test.pdf", "123", nDocumentID, 0)
```

Delphi 7.0

```
var
	nDocumentID : integer;
	nRes : integer;
...
try
	CoPDFXCview1.OpenDocument('C:\Test.pdf', '123', nDocumentID, 0);
except
	on ex : Exception do
	begin
		nRes := EOleException(ex).ErrorCode;
	end;
end;
```

See Also

IPDFXCview::OpenDocument

3.2 How to Print a Document?

To use examples below you should open a document and obtain the unique identifier of the opened document before printing (see How to Open a Document?).

To print directly (without standard print dialog displaying) you should specify the PXCVA_Flags:: PXCVA_NoUI flag in last (**Flags**) argument of the method.

C++ (with ATL)

```
LONG nDocumentID;
...
HRESULT hRes = spViewer->PrintDocument(nDocumentID, PXCVA_NoUI);
```

C#

```
int nDocumentID;
int nRes;
```

```
...
try
{
	axCoPDFXCview1.PrintDocument(nDocumentID, PXCVA_Flags.PXCVA_NoUI);
}
catch (Exception ex)
{
	nRes = System.Runtime.InteropServices.Marshal.GetHRForException(ex);
}
```

VB.NET

```
Dim nDocumentID As Integer
Dim nRes As Integer
...
Try
	AxCoPDFXCview1.PrintDocument(nDocumentID, PXCVA_Flags.PXCVA_NoUI);
Catch ex As Exception
	nRes = System.Runtime.InteropServices.Marshal.GetHRForException(ex)
End Try
```

VB 6.0

```
Dim nDocumentID As Long
...
Call CoPDFXCview1.PrintDocument(nDocumentID, PXCVA_Flags.PXCVA_NoUI);
```

Delphi 7.0

```
var
	nDocumentID : integer;
	nRes : integer;
...
try
	CoPDFXCview1.PrintDocument(nDocumentID, PXCVA_NoUI);
except
	on ex : Exception do
	begin
		nRes := EOleException(ex).ErrorCode;
	end;
end;
```

See Also

IPDFXCview::PrintDocument

3.3 How to Save a Document?

To use examples below you should open a document and obtain the unique identifier of the opened document before saving (see How to Open a Document?).

C++ (with ATL)

```
LONG nDocumentID;
...
HRESULT hRes = spViewer->SaveDocument(nDocumentID, NULL, 0, 0);
```

C#

```
int nDocumentID;
int nRes;
...
try
{
	axCoPDFXCview1.SaveDocument(nDocumentID, null, 0, 0);
}
catch (Exception ex)
{
	nRes = System.Runtime.InteropServices.Marshal.GetHRForException(ex);
}
```

VB.NET

```
Dim nDocumentID As Integer
Dim nRes As Integer
...
Try
	AxCoPDFXCview1.SaveDocument(nDocumentID, Nothing, 0, 0);
Catch ex As Exception
	nRes = System.Runtime.InteropServices.Marshal.GetHRForException(ex)
End Try
```

VB 6.0

```
Dim nDocumentID As Long
...
Call CoPDFXCview1.SaveDocument(nDocumentID, '', 0, 0);
```

Delphi 7.0

```
var
	nDocumentID : integer;
	nRes : integer;
...
try
	CoPDFXCview1.SaveDocument(nDocumentID, '', 0, 0);
except
	on ex : Exception do
	begin
		nRes := EOleException(ex).ErrorCode;
	end;
end;
```

See Also

IPDFXCview::SaveDocument

3.4 How to Close a Document?

To use examples below you should open a document and obtain the unique identifier of the opened document before closing (see How to Open a Document?).

C++ (with ATL)

```
LONG nDocumentID;
...
```

```
HRESULT hRes = spViewer->CloseDocument(nDocumentID, 0);
```

C#

```
int nDocumentID;
int nRes;
...
try
{
	axCoPDFXCview1.CloseDocument(nDocumentID, 0);
}
catch (Exception ex)
{
	nRes = System.Runtime.InteropServices.Marshal.GetHRForException(ex);
}
```

VB.NET

```
Dim nDocumentID As Integer
Dim nRes As Integer
...
Try
	AxCoPDFXCview1.CloseDocument(nDocumentID, 0);
Catch ex As Exception
	nRes = System.Runtime.InteropServices.Marshal.GetHRForException(ex)
End Try
```

VB 6.0

```
Dim nDocumentID As Long
...
Call CoPDFXCview1.CloseDocument(nDocumentID, 0);
```

Delphi 7.0

```
var
	nDocumentID : integer;
	nRes : integer;
...
try
	CoPDFXCview1.CloseDocument(nDocumentID, 0);
except
	on ex : Exception do
	begin
		nRes := EOleException(ex).ErrorCode;
	end;
end;
```

See Also

IPDFXCview::CloseDocument, IPDFXCview::CloseAllDocuments

3.5 How to Disable a Command?

Most commands in the Viewer may be accessed to enable or disable them by changing their **State** to *Online* or *Offline.*

To Disable the **"Print"** command, to prevent any printing from the Viewer, set the command as:

```
...
SetProperty("Commands["Print"].State", "Offline", 0);
```

See Also

Command List, Objects::Commands, Objects::Commands::<Item>

3.6 How to Enumerate Characters in Document?

To use examples below you should open a document and obtain the unique identifier of the opened document.

C++ (with ATL)

```
#define __S2BS(_Str_) CComBSTR(LPCWSTR(_Str_)).m_str
...
CComPtr<IPDFXCview> spView;
CComPtr<IPDFXCsmartp> spDoc;

CString str;
str.Format(L"Documents[#%d]", iDocID);
CComVariant dataIn, dataOut;
spView->DoVerb(__S2BS(str), __S2BS(L".SP"), dataIn, &dataOut, 0);
ATLASSERT(dataOut.vt == VT_DISPATCH);
spDoc = static_cast<IPDFXCsmartp*>(dataOut.punkVal);

dataOut = 0L;
spDoc->GetProperty(__S2BS(L"Pages.Count"), &dataOut, 0);
const LONG pagesCount = dataOut.lVal;
for (LONG i = 0; i < pagesCount; i++)
{
   CComPtr<IPDFXCsmartp> spPageChars;
   str.Format(L"Pages[%d].Chars", i);
   dataOut = 0L;
   spDoc->GetProperty(__S2BS(str), &dataOut, 0);
   const LONG charsCount = dataOut.lVal;
   for (LONG j = 0; j < charsCount; j++)
   {
      CComPtr<IPDFXCsmartp> spChar;
      spPageChars->GetItemPointByIndex(j, &spChar.p);
      spChar->GetProperty(__S2BS(L"Code"), &dataOut, 0);
      WCHAR ch = (WCHAR)dataOut.lVal;
      ...
   }
}
```

C#

```
PDFXCviewAxLib.IPDFXCsmartp spDoc;
object dataIn = null;
object dataOut = null;
axCoPDFXCview1.DoVerb("Documents[#" + iDocID.ToString() + "]", ".SP",
                      dataIn, out dataOut, 0);
spDoc = (PDFXCviewAxLib.IPDFXCsmartp)dataOut;
dataOut = 0;
spDoc.GetProperty("Pages.Count", out dataOut, 0);
int pagesCount = (int)dataOut;
for (int i = 0; i < pagesCount; i++)
```

```
{
   PDFXCviewAxLib.IPDFXCsmartp spPageChars;
   spDoc.GetSubPoint("Pages[" + i.ToString() + "].Text.Chars", out
spPageChars);
   dataOut = 0;
   spPageChars.GetProperty("Count", out dataOut, 0);
   int charsCount = (int)dataOut;
   for (int j = 0; j < charsCount; j++)
   {
      PDFXCviewAxLib.IPDFXCsmartp spChar;
      spPageChars.GetItemPointByIndex(j, out spChar);
      if (spChar == null)
         continue;
      dataOut = 0;
      spChar.GetProperty("Code", out dataOut, 0);
      Char ch = (Char)(int)dataOut;
      ...
      System.Runtime.InteropServices.Marshal.ReleaseComObject(spChar);
   }
   System.Runtime.InteropServices.Marshal.ReleaseComObject(spPageChars);
}
System.Runtime.InteropServices.Marshal.ReleaseComObject(spDoc);
```

VB 6.0

```
Dim dataIn As Variant
Dim dataOut As Variant
Dim spDoc As IPDFXCsmartp
Dim pagesCount As Integer
Dim objName As String

objName = "Documents[#" & iDocID & "]"
dataIn = 0
Call CoPDFXCview1.DoVerb(objName, ".SP", dataIn, dataOut, 0)
Set spDoc = dataOut

dataOut = 0
Call spDoc.GetProperty("Pages.Count", dataOut, 0)

pagesCount = dataOut

Dim i As Integer
For i = 0 To pagesCount - 1
  Dim spPageChars As IPDFXCsmartp
  Call spDoc.GetSubPoint("Pages[" & i & "].Text.Chars", spPageChars)
  dataOut = 0
  Call spPageChars.GetProperty("Count", dataOut, 0)
  Dim charsCount As Integer
  charsCount = dataOut
  If charsCount > 0 Then
      Dim j As Long
      For j = 0 To charsCount - 1
          Dim spChar As IPDFXCsmartp
          Call spPageChars.GetItemPointByIndex(j, spChar)
          If IsNull(spChar) = False Then
            Call spChar.GetProperty("Code", dataOut, 0)
            Dim ch As String
            ch = ChrW(dataOut)
            ...
          End If
      Next j
```

```
  End If
Next i
```

Delphi

```
var
  i,j:integer;
  spDoc, spPageChars, spChar: IPDFXCsmartp;
  dataIn, dataOut:OleVariant;
  pagesCount, charsCount:integer;
  iDocID:integer;
  ch: WideChar;
begin
  ...
  CoPDFXCview1.DoVerb('Documents[#'+inttostr(iDocID)+']', '.SP', dataIn,
dataOut, 0);
  spDoc := IDispatch(dataOut) as IPDFXCsmartp;
  spDoc.GetProperty('Pages.Count', dataOut, 0);
  pagesCount := dataOut;
  for i := 0 to pagesCount - 1 do
  begin
    spDoc.GetSubPoint('Pages['+inttostr(i)+'].Text.Chars', spPageChars);
    dataOut := 0;
    spPageChars.GetProperty('Count', dataOut, 0);
    charsCount := dataOut;
    for j := 0 to charsCount - 1 do
    begin
      spPageChars.GetItemPointByIndex(j, spChar);
      if (spChar = nil) then
         continue;
      spChar.GetProperty('Code', dataOut, 0);
      ch := WideChar(integer(dataOut));
      ...
    end;
  end;
end;
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.Pages::<Item>.Text,
IPDFXCsmartp

3.7 How to Extract Text from Document?

To use examples below you should open a document and obtain the unique identifier of the opened document.

C++ (with ATL)

```
#define __S2BS(_Str_) CComBSTR(LPCWSTR(_Str_)).m_str
...
CString sPdfText;
CComVariant dataIn, dataOut;
CString str;
str.Format(L"Documents[#%d]", iDocID);

m_spView->DoVerb(__S2BS(str), __S2BS(L"GetAllText"), dataIn, &dataOut, 0);
sPdfText = dataOut.bstrVal;
```

C#

```
String sPdfText;
object dataOut = null;

axCoPDFXCview1.DoVerb("Documents[#" + iDocID.ToString() + "]", "GetAllText",
                      null, out dataOut, 0);
sPdfText = (String)dataOut;
```

VB 6.0

```
Dim sPdfText As String
Dim dataOut As Variant

Call CoPDFXCview1.DoVerb("Documents[#" & iDocID & "]", "GetAllText",
                         Nothing, dataOut, 0)
sPdfText = dataOut
```

Delphi

```
var
  dataIn, dataOut:OleVariant;
  iDocID:integer;
  sPdfText:string;
begin
  ...
  CoPDFXCview1.DoVerb('Documents[#' + IntToStr(iDocID) + ']', 'GetAllText',
                      dataIn, dataOut, 0);
  sPdfText := dataOut;
  ...
end;
```

See Also

IPDFXCview::DoVerb,
Objects::Documents::<Item>.GetAllText

3.8 How to Summarize Annotations from Document?

This example shows how you can easily save all comments from PDF document to the text file without showing prompt dialog. To use this example you should open document first.

C#

```
object dataIn, dataOut;
dataIn = null;
axCoPDFXCview1.SetProperty("Operations.SummarizeAnnots.Output.Type", "txt");
axCoPDFXCview1.SetProperty("Operations.SummarizeAnnots.Output.AutoView",
"false");
axCoPDFXCview1.SetProperty("Operations.SummarizeAnnots.Output.TXT.
FolderName", "C:\\");
axCoPDFXCview1.SetProperty("Operations.SummarizeAnnots.Output.TXT.FileName",
                           "CommentsFromPDF");
axCoPDFXCview1.DoVerb("Documents[0]", "SummarizeAnnots", dataIn, out
dataOut,
                      (int)PDFXCviewAxLib.PXCVA_Flags.PXCVA_NoUI);
```

VB 6.0

```
Call CoPDFXCview1.SetProperty("Operations.SummarizeAnnots.Output.Type",
"txt")
Call CoPDFXCview1.SetProperty("Operations.SummarizeAnnots.Output.AutoView",
"false")
Call CoPDFXCview1.SetProperty("Operations.SummarizeAnnots.Output.TXT.
FolderName", "C:\\")
Call CoPDFXCview1.SetProperty("Operations.SummarizeAnnots.Output.TXT.
FileName",
                              "CommentsFromPDF")
Call CoPDFXCview1.DoVerb("Documents[0]", "SummarizeAnnots", Nothing,
Nothing,
                         PXCVA_Flags.PXCVA_NoUI)
```

Delphi

```
var
  dataOut:OleVariant;
  ....
begin
  ....
  CoPDFXCview1.SetProperty('Operations.SummarizeAnnots.Output.Type', 'txt',
0);
  CoPDFXCview1.SetProperty('Operations.SummarizeAnnots.Output.AutoView',
'false', 0);
  CoPDFXCview1.SetProperty('Operations.SummarizeAnnots.Output.TXT.
FolderName', 'C:\', 0);
  CoPDFXCview1.SetProperty('Operations.SummarizeAnnots.Output.TXT.FileName',
                           'CommentsFromPDF', 0);
  CoPDFXCview1.DoVerb('Documents[0]', 'SummarizeAnnots', '', dataOut,
PXCVA_NoUI);
  ....
end;
```

See Also

IPDFXCview::DoVerb,
Objects::Operations.SummarizeAnnots

4 Tracker Software Products

Who are we and what do we do?

We at Tracker Software Products take great pride in the software products we create and distribute. We sell our products directly, via Distributors, Resellers and OEM partners - in some cases with our products and larger partners these products are sold under different labels and names than those we sell directly, this is to allow our partners to build a following for their own brand and protect their future.

However - no matter how our products reach you, we want you to experience the best results possible - please do contact us if for any reason you are dissatisfied or have a suggestion how we can improve our product offerings.

You may also be interested in related products available from us - in the following brief topics you will find details of how to contact us, request support and summary details of the products available from us for both 'End Users' and Software Development tools for other Software developers to utilise in their product offerings.

Please do contact us if you cannot find the information you require within this manual/help file.

4.1 Contact Us

How to Contact Us...

North/South America, Australia, Asia:
Tracker Software Products (Canada) Ltd.,
3 - 466 Trans Canada Highway
Duncan. BC
V9L 3R6, Canada
Sales & Admin:
Tel: Canada (+00) 1-250-597-1621
Fax: Canada (+00) 1-250-597-1623

European Office:
7 Beech Gardens
Crawley Down., RH10 4JB
Sussex, United Kingdom
Sales:
Tel: +44 (0) 20 8555 1122
Fax: +001 250 -597-1623

Our Web Site: http://www.tracker-software.com
We also have offices and representatives in several other locations including: United States, France, Germany and Ukraine - in some instances after an initial contact with our UK office you may be referred to one of these locations if appropriate.

To contact us for support related issues:
Please see this FAQ page before contacting our support department - it may save you the task!
We recommend you use our Web Based User Support Forums and scan the existing library of questions and answers, if you don't find a suitable response then feel free to post your own - all questions receive an answer within 1 business day at worst!
If for any reason you have difficulty linking to the forum or feel it is inappropriate for your needs then please email sales@tracker-software.com, we regret we cannot answer support requests via telephone without a valid support contract. The number above is answered by administration staff who are not trained to assist with technical problems.

To Contact us for Sales/Administration related issues:
sales@tracker-software.com End User, Developer and OEM.
admin@tracker-software.com
All this information and a good deal more is available via our web site and the links provided.

Magazine reviews and press requests.
We are keen to assist in any way possible - please contact our sales department for any information or help you may require.

4.2 Products

Products Offered By Tracker Software Products Updates Can be downloaded from our Update's page at our Web site.

End User/Retail Products

You can Purchase Direct from our web site and be using any of our products the same day!

- PDF-XChange - Create fully native Adobe compatible PDF Files from virtually any Windows 32 Bit software application.
- PDF-XChange Viewer (Free and PRO Versions) - View, Mark-up and Manipulate PDF files and much more!
- PDF-Tools - Create and manipulate Adobe PDF Files and batch Convert Images to PDF Files and more...

- Raster-XChange - Create and manipulate Image files from any Windows document or application.
- TIFF-XChange - Create and manipulate TIFF files from any Windows document or application.

Software Developers SDK's and other Products

You can Purchase Developer SDK's from our web site and be using any of our products the same day!

- PDF-XChange - Create fully native Adobe compatible PDF Files from your application output.
- PDF-Tools - Create and manipulate Adobe PDF Files and batch Convert Images to PDF Files and more...
- PDF-XChange Viewer SDK - Embed PDF viewing directly within your software applications.
- Image-XChange SDK - Print, Convert, Scan and View Imaging formats!
- Raster-XChange SDK - Create and manipulate Image files from any Windows document or application.
- TIFF-XChange SDK - Create and manipulate TIFF files from any Windows document or application.
- OCR-XChange - Add Optical Character Recognition functionality to your applications - coming late 2008.

Trial Versions

All of our products are available as fully functional evaluation downloads for you to try before you buy - usually printing a demo watermark/stamp to differentiate between output created with the evaluation or licensed versions. We recommend that all users test the product they wish to buy first - thus ensuring you only buy when you are satisfied that the product meets your needs.
Trial versions are available from our web site:
For more details visit our web site or contact us by email.

Index

- C -

Commands List 59
Contact 298
Contact Us 298

- O -

Other Products 298

- T -

Tracker Software Products 298