

Version 9.5.0.0

# Foxit Reader PDF API Reference

For developers

©2014 Foxit Corporation. All Rights Reserved

**Copyright © 2010 Foxit Corporation. All Rights Reserved.**

No part of this document can be reproduced, transferred, distributed or stored in any format without the prior written permission of Foxit.

Anti-Grain Geometry - Version 2.3, Copyright (C) 2002-2005 Maxim Shemanarev (<http://www.antigrain.com>). FreeType2 (freetype2.2.1), Copyright (C) 1996-2001, 2002, 2003, 2004| David Turner, Robert Wilhelm, and Werner Lemberg. LibJPEG (jpeg V6b 27- Mar-1998), Copyright (C) 1991-1998 Independent JPEG Group. ZLib (zlib 1.2.2), Copyright (C) 1995-2003 Jean-loup Gailly and Mark Adler . Little CMS, Copyright (C) 1998-2004 Marti Maria. Kakadu, Copyright (C) 2001, David Taubman, The University of New South Wales (UNSW). PNG, Copyright (C) 1998-2009 Glenn Randers-Pehrson. LibTIFF, Copyright (C) 1988-1997 Sam Leffler and Copyright (C) 1991-1997 Silicon Graphics, Inc.

Permission to copy, use, modify, sell and distribute this software is granted provided this copyright notice appears in all copies. This software is provided "as is" without express or implied warranty, and with no claim as to its suitability for any purpose.

# Content

Content.....	3
Preface.....	24
What's in this guide? .....	24
Who should read this guide?.....	24
Related documentation.....	24
Support Layer.....	25
Object List.....	25
General.....	28
Description.....	28
Definitions.....	28
Typedefs.....	35
Callbacks.....	43
Structures .....	67
FS_AffineMatrix.....	71
Description.....	71
Functions.....	73
FS_Base64Decoder.....	83
Description.....	83
Callbacks.....	84
Functions.....	84
FS_Base64Encoder .....	89
Description.....	89
Callbacks.....	89
Functions.....	90
FS_BinaryBuf .....	94
Description.....	94
Functions.....	95
FS_ByteArray .....	105
Description.....	105
Functions.....	105
FS_ByteString.....	115
Description.....	115
Functions.....	118
FS_ByteStringArray .....	145
Description.....	145
Functions.....	146
FS_CharMap .....	150
Description.....	150
Functions.....	151
FS_CodeTransformation.....	153
Description.....	153
Functions.....	153
FS_DIBAttribute.....	164
Description.....	164
Definitions.....	164
Enumerations .....	168
FS_DIBitmap .....	170
Description.....	170

Definitions.....	172
Enumerations .....	183
Typedefs.....	184
Functions.....	185
FS_DWordArray.....	221
Description.....	221
Functions.....	222
FS_ExtensionHFTMgr.....	231
Description.....	231
Functions.....	231
FS_FileReadHandler.....	235
Description.....	235
Callbacks.....	235
Functions.....	237
FS_FileStream.....	240
Description.....	240
Definitions.....	241
Callbacks.....	242
Functions.....	248
FS_FileWriteHandler.....	250
Description.....	250
Callbacks.....	251
Functions.....	253
FS_FloatRectArray.....	254
Description.....	254
Functions.....	255
FS_GUID .....	264
Description.....	264
Functions.....	264
FS_HFT.....	265
Description.....	265
FS_MapByteStringToPtr .....	266
Description.....	266
Functions.....	266
FS_MapPtrToPtr .....	275
Description.....	275
Functions.....	275
FS_PauseHandler.....	283
Description.....	283
Callbacks.....	283
Functions.....	284
FS_PtrArray .....	285
Description.....	285
Functions.....	286
FS_StreamWriteHandler.....	295
Description.....	295
Callbacks.....	295
Functions.....	297
FS_UTF8Decoder.....	298
Description.....	299

Functions.....	299
FS_UTF8Encoder .....	303
Description.....	303
Functions.....	303
FS_UUID .....	307
Description.....	307
Functions.....	307
FS_WideString.....	311
Description.....	311
Functions.....	314
FS_WideStringArray .....	342
Description.....	342
Functions.....	343
FS_WordArray.....	347
Description.....	347
Functions.....	347
FS_XMLElement.....	357
Description.....	357
Enumerations .....	358
Functions.....	358
Reader Layer.....	389
Object List.....	389
General.....	397
Description.....	397
Definitions.....	397
Enumerations .....	421
Callbacks.....	429
Structures .....	682
FDRM_CategoryRead .....	694
Description.....	694
Functions.....	694
FDRM_CategoryWrite .....	696
Description.....	696
FDRM_DescData.....	696
Description.....	696
FDRM_DescRead.....	696
Description.....	696
FDRM_DescWrite .....	696
Description.....	696
FDRM_EncryptDictRead .....	696
Description.....	696
FDRM_Encryptor .....	696
Description.....	696
FDRM_EnvelopeRead.....	696
Description.....	696
FDRM_FoacRead .....	696
Description.....	696
FDRM_FoacWrite .....	696
Description.....	697
FDRM_Mgr .....	697

Description.....	697
FDRM_PDFSchema .....	697
Description.....	697
FDRM_PKI.....	697
Description.....	697
FDRM_PresentationData.....	697
Description.....	697
FDRM_ScriptData .....	697
Description.....	697
FDRM_SignatureData .....	697
Description.....	697
FOFD_Action .....	697
Description.....	697
FOFD_Actions .....	697
Description.....	697
FOFD_CodeC .....	697
Description.....	697
FOFD_Creator .....	697
Description.....	697
FOFD_CryptoDict .....	697
Description.....	698
FOFD_Dest .....	698
Description.....	698
FOFD_Doc .....	698
Description.....	698
FOFD_Outline .....	698
Description.....	698
FOFD_Package .....	698
Description.....	698
FOFD_Page.....	698
Description.....	698
FOFD_Parser .....	698
Description.....	698
FOFD_Perms .....	698
Description.....	698
FOFD_PrintSetting .....	698
Description.....	698
FOFD_Sign .....	698
Description.....	698
FOFD_SM4CryptoHandler.....	698
Description.....	698
FOFD_SMSecurityHandler .....	698
Description.....	699
FOFD_StdCertSecurityHandler.....	699
Description.....	699
FOFD_StdCryptoHandler .....	699
Description.....	699
FOFD_StdSecurityHandler.....	699
Description.....	699
FOFD_Sys.....	699

---

Description.....	699
FOFD_UIMgr .....	699
Description.....	699
FOFD_VPrefers .....	699
Description.....	699
FOFD_WriteDoc.....	699
Description.....	699
FR_ActionWizardData .....	699
Description.....	699
FR_ActionWizardHandler .....	699
Description.....	699
Definitions.....	699
Enumerations .....	701
Callbacks.....	701
FR_Annot.....	702
Description.....	702
Functions.....	702
FR_App.....	705
Description.....	705
Enumerations .....	706
Callbacks.....	706
Functions.....	710
FR_BulbMsgCenter .....	820
Description.....	821
Functions.....	821
FR_CloudLoginProvider.....	830
Description.....	830
Callbacks.....	830
Structures .....	833
Functions.....	834
FR_Cursor.....	839
Description.....	839
Definitions.....	839
FR_CustomSignature .....	842
Description.....	842
Functions.....	843
FR_Document .....	848
Description.....	848
Definitions.....	851
Enumerations .....	853
Callbacks.....	855
Functions.....	858
FR_DocView .....	925
Description.....	925
Definitions.....	926
Enumerations .....	928
Structures .....	930
Functions.....	932
FR>Edit .....	955
Description.....	955

Definitions.....	955
Callbacks.....	957
Structures .....	958
FR_Edit_FontMap .....	958
Description.....	958
Enumerations .....	958
FR_Edit_Iterator .....	959
Description.....	959
FR_Edit_UndoItem.....	959
Description.....	959
FR_EmptyFrameWndViewEventHandler .....	959
Description.....	959
Callbacks.....	959
FR_FormatTools .....	967
Description.....	967
Definitions.....	969
Enumerations .....	977
Callbacks.....	980
Functions.....	998
FR_FuncBtn.....	1033
Description.....	1033
Functions.....	1033
FR_HtmlMgr .....	1043
Description.....	1043
Callbacks.....	1043
Functions.....	1050
FR_Language .....	1058
Description.....	1059
Functions.....	1059
FR_Menu .....	1067
Description.....	1067
Definitions.....	1068
Functions.....	1070
FR_MenuBar.....	1081
Description.....	1081
Functions.....	1082
FR_MenuItem .....	1086
Description.....	1086
Functions.....	1087
FR_MenuOwnerDrawHandler.....	1103
Description.....	1103
Callbacks.....	1103
FR_MessageBar .....	1105
Description.....	1106
Enumerations .....	1106
Functions.....	1107
FR_PageView .....	1123
Description.....	1123
Enumerations .....	1124
Structures .....	1125

---

Functions.....	1126
FR_PanelMgr.....	1144
Description.....	1144
Definitions.....	1145
Callbacks.....	1146
Functions.....	1149
FR_ProfStore .....	1170
Description.....	1170
FR_PropertyTools.....	1170
Description.....	1170
Callbacks.....	1171
Functions.....	1174
FR_ResourcePropertyBox .....	1179
Description.....	1179
Definitions.....	1180
Callbacks.....	1182
Functions.....	1197
FR_ResourcePropertyPage .....	1207
Description.....	1207
FR_ResourcePropertySource .....	1208
Description.....	1208
FR_RibbonBackStageViewItem.....	1208
Description.....	1208
Enumerations .....	1209
Callbacks.....	1210
Functions.....	1212
FR_RibbonBar .....	1227
Description.....	1227
Callbacks.....	1227
Functions.....	1230
FR_RibbonButton .....	1250
Description.....	1250
Functions.....	1250
FR_RibbonCategory .....	1254
Description.....	1254
Callbacks.....	1255
Functions.....	1256
FR_RibbonCheckBox .....	1273
Description.....	1273
Functions.....	1273
FR_RibbonColorButton.....	1275
Description.....	1275
Functions.....	1276
FR_RibbonComboBox .....	1292
Description.....	1292
Functions.....	1293
FR_RibbonEdit .....	1304
Description.....	1304
Functions.....	1304
FR_RibbonElement.....	1311

Description.....	1311
Enumerations .....	1312
Functions.....	1313
<b>FR_RibbonFontComboBox .....</b>	<b>1347</b>
Description.....	1347
Functions.....	1347
<b>FR_RibbonLabel.....</b>	<b>1361</b>
Description.....	1361
Functions.....	1361
<b>FR_RibbonListButton.....</b>	<b>1362</b>
Description.....	1362
Enumerations .....	1362
Functions.....	1363
<b>FR_RibbonPaletteButton .....</b>	<b>1374</b>
Description.....	1374
Functions.....	1375
<b>FR_RibbonPanel .....</b>	<b>1397</b>
Description.....	1397
Callbacks.....	1397
Functions.....	1399
<b>FR_RibbonRadioButton .....</b>	<b>1413</b>
Description.....	1414
Functions.....	1414
<b>FR_RibbonSlider .....</b>	<b>1416</b>
Description.....	1416
Functions.....	1416
<b>FR_RibbonStyleButton.....</b>	<b>1423</b>
Description.....	1423
Functions.....	1423
<b>FR_RibbonStyleListBox.....</b>	<b>1428</b>
Description.....	1428
Callbacks.....	1429
Functions.....	1430
<b>FR_RibbonStyleStatic.....</b>	<b>1444</b>
Description.....	1444
Functions.....	1444
<b>FR_ScrollBarThumbnailView .....</b>	<b>1450</b>
Description.....	1450
Functions.....	1450
<b>FR_SpellCheck .....</b>	<b>1452</b>
Description.....	1452
Functions.....	1452
<b>FR_StatusBar .....</b>	<b>1457</b>
Description.....	1457
Functions.....	1457
<b>FR_Sys .....</b>	<b>1465</b>
Description.....	1465
Functions.....	1465
<b>FR_TabBand .....</b>	<b>1471</b>
Description.....	1471

---

Callbacks.....	1471
Functions.....	1473
FR_TextSelectTool.....	1480
Description.....	1480
Functions.....	1480
FR_ThumbnailView .....	1489
Description.....	1489
Functions.....	1489
FR_Tool .....	1492
Description.....	1492
Definitions.....	1494
Callbacks.....	1510
Functions.....	1525
FRToolBar .....	1528
Description.....	1528
Definitions.....	1529
Functions.....	1539
FRToolButton .....	1553
Description.....	1554
Callbacks.....	1554
Functions.....	1555
FRTouchup .....	1567
Description.....	1567
Definitions.....	1567
Structures .....	1571
FRUIProgress.....	1572
Description.....	1572
Functions.....	1573
FRVariableText.....	1578
Description.....	1578
FRVariableText_Iterator.....	1578
Description.....	1579
FRVariableText_Provider.....	1579
Description.....	1579
FRVTLine .....	1579
Description.....	1579
FRVTSecProps .....	1579
Description.....	1579
FRVTSection .....	1579
Description.....	1579
FRVTWord .....	1579
Description.....	1579
FRVTWordPlace.....	1579
Description.....	1579
FRVTWordProps .....	1579
Description.....	1579
FRVTWordRange .....	1579
Description.....	1579
FRWindowsDIB .....	1579
Description.....	1579

Functions.....	1579
FRMS_Common.....	1588
Description.....	1588
FRMS_Decryption.....	1588
Description.....	1588
FRMS_Encryption.....	1588
Description.....	1588
PD Layer .....	1588
Object List.....	1588
General.....	1601
Description.....	1601
Definitions.....	1601
Typedefs.....	1607
Callbacks.....	1607
FDF_Document.....	1609
Description.....	1609
Functions.....	1610
FDRM_PDFSecurityHandler.....	1626
Description.....	1626
FOFD_ACTION .....	1626
Description.....	1626
FOFD_ActionArea.....	1626
Description.....	1626
FOFD_ActionGoto .....	1626
Description.....	1626
FOFD_ActionGotoA.....	1626
Description.....	1626
FOFD_ActionRegion.....	1626
Description.....	1626
FOFD_ACTIONS .....	1626
Description.....	1626
FOFD_ANNOT .....	1626
Description.....	1626
FOFD_Bookmark .....	1626
Description.....	1626
FOFD_Bookmarks.....	1626
Description.....	1626
FOFD_CREATOR.....	1627
Description.....	1627
FOFD_CRYPTODICT .....	1627
Description.....	1627
FOFD_CryptoHandler .....	1627
Description.....	1627
FOFD_DEST .....	1627
Description.....	1627
FOFD_DIBAttribute .....	1627
Description.....	1627
FOFD_Document.....	1627
Description.....	1627
FOFD_DriverDevice.....	1627

---

Description.....	1627
FOFD_FILEPACKAGE .....	1627
Description.....	1627
FOFD_FileStream.....	1627
Description.....	1627
FOFD_OBJECT.....	1627
Description.....	1627
FOFD_OUTLINE.....	1628
Description.....	1628
FOFD_PAGE.....	1628
Description.....	1628
FOFD_PARSER .....	1628
Description.....	1628
FOFD_Path .....	1628
Description.....	1628
FOFD_PauseHandler .....	1628
Description.....	1628
Functions.....	1628
FOFD_PERMISSIONS .....	1629
Description.....	1629
FOFD_ProgressiveRenderer .....	1629
Description.....	1630
FOFD_RenderContext .....	1630
Description.....	1630
FOFD_RenderDevice .....	1630
Description.....	1630
FOFD_RenderOptions .....	1630
Description.....	1630
FOFD_SecurityHandler .....	1630
Description.....	1630
FOFD_SIGNATURE.....	1630
Description.....	1630
FOFD_VPREFERENCES .....	1630
Description.....	1630
FOFD_WRITEPAGE .....	1630
Description.....	1630
FOFD_WRITESIGNATURE .....	1630
Description.....	1630
FPD_AAction .....	1630
Description.....	1631
Enumerations .....	1631
Functions.....	1633
FPD_Action .....	1638
Description.....	1638
Definitions.....	1640
Enumerations .....	1642
Functions.....	1643
FPD_ActionFields.....	1673
Description.....	1673
Functions.....	1673

---

FPD_AllStates.....	1678
Description.....	1678
FPD_Annot .....	1678
Description.....	1678
Definitions.....	1679
Enumerations .....	1682
Functions.....	1683
FPD_AnnotList .....	1690
Description.....	1691
Functions.....	1691
FPD_Array .....	1703
Description.....	1703
Functions.....	1703
FPD_BackgroundDrawHandler.....	1718
Description.....	1718
Callbacks.....	1719
FPD_Bookmark .....	1720
Description.....	1720
Definitions.....	1720
Functions.....	1721
FPD_Boolean.....	1727
Description.....	1727
Functions.....	1727
FPD_CIDFont .....	1728
Description.....	1728
Functions.....	1729
FPD_CIDUtil .....	1738
Description.....	1738
Functions.....	1738
FPD_ClipPath .....	1739
Description.....	1739
Functions.....	1740
FPD_Color .....	1748
Description.....	1748
Functions.....	1750
FPD_ColorSpace.....	1757
Description.....	1758
Functions.....	1758
FPD_ColorState .....	1769
Description.....	1769
Functions.....	1770
FPD_ConnectedInfo .....	1776
Description.....	1776
Definitions.....	1777
Functions.....	1781
FPD_ContentGenerator.....	1793
Description.....	1793
Enumerations .....	1793
FPD_ContentMark .....	1794
Description.....	1794

---

Functions.....	1795
FPD_ContentMarkItem.....	1801
Description.....	1801
Enumerations .....	1802
Functions.....	1802
FPD_Creator .....	1806
Description.....	1806
Definitions.....	1807
Functions.....	1808
FPD_CreatorOption .....	1822
Description.....	1822
Callbacks.....	1822
FPD_CryptoHandler .....	1824
Description.....	1824
FPD_DefaultAppearance .....	1824
Description.....	1824
Functions.....	1825
FPD_Dest.....	1834
Description.....	1834
Definitions.....	1834
Functions.....	1836
FPD_Dictionary .....	1840
Description.....	1840
Functions.....	1840
FPD_DocJSActions .....	1861
Description.....	1861
Functions.....	1862
FPD_Document.....	1867
Description.....	1867
Definitions.....	1870
Functions.....	1877
FPD_EnumPageHandler .....	1915
Description.....	1915
Callbacks.....	1915
FPD_EPUB .....	1916
Description.....	1916
FPD_FileSpec .....	1916
Description.....	1916
Functions.....	1917
FPD_Font.....	1921
Description.....	1921
Definitions.....	1924
Functions.....	1935
FPD_FontEncoding.....	1964
Description.....	1964
Functions.....	1965
FPD_Form.....	1969
Description.....	1970
Functions.....	1973
FPD_FormControl .....	1995

---

Description.....	1995
Definitions.....	1996
Functions.....	1999
FPD_FormField .....	2028
Description.....	2029
Definitions.....	2030
Enumerations .....	2036
Functions.....	2039
FPD_FormNotify .....	2073
Description.....	2074
Callbacks.....	2074
Functions.....	2080
FPD_FormObject.....	2081
Description.....	2081
Functions.....	2081
FPD_FT_Face .....	2085
Description.....	2085
FPD_Function .....	2085
Description.....	2085
FPD_FXFont.....	2086
Description.....	2086
FPD_FXFontEncoding .....	2086
Description.....	2086
Definitions.....	2087
Functions.....	2087
FPD_FxgeDevice.....	2092
Description.....	2092
Functions.....	2092
FPD_GeneralState.....	2094
Description.....	2094
Functions.....	2095
FPD_GraphState .....	2102
Description.....	2102
Enumerations .....	2102
Functions.....	2104
FPD_IconFit.....	2113
Description.....	2113
Functions.....	2113
FPD_Image .....	2119
Description.....	2119
Structures .....	2120
Functions.....	2121
FPD_ImageObject.....	2132
Description.....	2132
Functions.....	2132
FPD_InlineImages .....	2136
Description.....	2136
Functions.....	2137
FPD_InterForm .....	2140
Description.....	2140

---

Definitions.....	2142
Functions.....	2143
FPD_Link.....	2191
Description.....	2191
Functions.....	2192
FPD_LinkExtract .....	2196
Description.....	2197
Functions.....	2197
FPD_LWinParam.....	2200
Description.....	2200
Functions.....	2201
FPD_MediaPlayer.....	2207
Description.....	2207
Enumerations .....	2207
Functions.....	2209
FPD_MeshStream.....	2212
Description.....	2213
Functions.....	2213
FPD_Name.....	2217
Description.....	2217
Functions.....	2218
FPD_NameTree .....	2220
Description.....	2220
Functions.....	2220
FPD_Null .....	2226
Description.....	2226
Functions.....	2227
FPD_Number .....	2227
Description.....	2227
Functions.....	2227
FPD_ObjArchiveLoader .....	2234
Description.....	2234
Functions.....	2234
FPD_ObjArchiveSaver .....	2241
Description.....	2241
Functions.....	2241
FPD_Object.....	2249
Description.....	2249
Definitions.....	2254
Functions.....	2258
FPD_OCCContext .....	2269
Description.....	2270
Enumerations .....	2270
Functions.....	2271
FPD_OCCContextHandler .....	2275
Description.....	2275
Callbacks.....	2276
FPD_OCGroup .....	2277
Description.....	2277
Callbacks.....	2278

Functions.....	2278
FPD_OCGroupSet .....	2286
Description.....	2286
Functions.....	2287
FPD_OCNotify .....	2291
Description.....	2291
Functions.....	2291
FPD_OCPatterns .....	2293
Description.....	2293
Functions.....	2293
FPD_Page .....	2300
Description.....	2300
Functions.....	2304
FPD_PageArchiveLoader .....	2330
Description.....	2331
Functions.....	2331
FPD_PageArchiveSaver .....	2335
Description.....	2336
Functions.....	2336
FPD_PageObject.....	2340
Description.....	2340
Definitions.....	2342
Structures .....	2346
Functions.....	2347
FPD_PageRenderCache .....	2360
Description.....	2360
Functions.....	2361
FPD_ParseOptions .....	2365
Description.....	2365
Functions.....	2365
FPD_Parser .....	2369
Description.....	2369
Definitions.....	2370
Structures .....	2373
Functions.....	2374
FPD_Path .....	2405
Description.....	2405
Functions.....	2405
FPD_PathObject .....	2414
Description.....	2414
Functions.....	2415
FPD_Pattern .....	2421
Description.....	2421
Functions.....	2421
FPD_ProgressiveEncryptHandler .....	2424
Description.....	2424
Callbacks.....	2424
FPD_ProgressiveRender .....	2426
Description.....	2426
Functions.....	2426

---

FPD_ProgressiveSearch.....	2429
Description.....	2430
Definitions.....	2430
Functions.....	2433
FPD_Reference.....	2438
Description.....	2438
Functions.....	2438
FPD_RenderContext.....	2442
Description.....	2442
Functions.....	2442
FPD_RenderDevice .....	2452
Description.....	2452
Functions.....	2453
FPD_RenderOptions .....	2475
Description.....	2475
Definitions.....	2476
Functions.....	2480
FPD_Rendition .....	2488
Description.....	2488
Functions.....	2490
FPD_ShadingObject .....	2522
Description.....	2522
Functions.....	2522
FPD_ShadingPattern.....	2527
Description.....	2527
Functions.....	2528
FPD_Stream.....	2534
Description.....	2534
Functions.....	2535
FPD_StreamAcc .....	2540
Description.....	2540
Functions.....	2540
FPD_StreamFilter .....	2545
Description.....	2546
Functions.....	2546
FPD_StreamWrite.....	2548
Description.....	2548
FPD_String .....	2548
Description.....	2548
Functions.....	2548
FPD_SubstFont .....	2552
Description.....	2552
FPD_TextObject .....	2552
Description.....	2552
Functions.....	2552
FPD_TextPage .....	2565
Description.....	2565
Definitions.....	2565
Structures .....	2569
Functions.....	2570

---

FPD_TextPageFind.....	2580
Description.....	2581
Functions.....	2581
FPD_TextState.....	2585
Description.....	2585
Functions.....	2586
FPD_TilingPattern.....	2597
Description.....	2597
Functions.....	2598
FPD_TrueTypeFont.....	2604
Description.....	2604
Functions.....	2604
FPD_Type1Font.....	2608
Description.....	2608
Functions.....	2608
FPD_Type3Char.....	2612
Description.....	2612
Functions.....	2613
FPD_Type3Font.....	2617
Description.....	2617
Functions.....	2617
FPD_UnencryptedWrapperCreator.....	2623
Description.....	2623
Functions.....	2623
FPD_WindowsDevice.....	2628
Description.....	2628
Functions.....	2628
FPD_Wrapper20Creator.....	2629
Description.....	2629
FPD_WrapperCreator.....	2629
Description.....	2629
Functions.....	2630
FPD_WrapperDoc.....	2633
Description.....	2633
Definitions.....	2634
Functions.....	2635
FRMS_Template.....	2640
Description.....	2640
Definition Groups .....	2640
AddHFTErrs .....	2640
CALL_REPLACE PROC .....	2640
FSBoolean.....	2641
FSBoolean.....	2641
FSFormatingFlags.....	2641
FSNull .....	2641
FSUUIDType.....	2641
Handshakeverssion .....	2642
INIT_CALLBACK_STRUCT.....	2642
SDKVersion.....	2642
FSDIBEXIFTAG .....	2642

FSDIBConvertFlags.....	2643
FSDIBBlendTypes.....	2643
FSDIBStretchFlags .....	2644
FSFillingModeFlags .....	2644
FS FileMode.....	2644
CaptureType.....	2644
DocPermission .....	2644
FRCIPHERFlag .....	2645
FRDataCollectionActionNames .....	2645
FRDataCollectionFunctionNames .....	2645
FROEMVersion .....	2646
FRSIGShowAPFlags .....	2646
FRSIGVerifySignatureStates .....	2646
SelectionType .....	2647
TaskPaneName .....	2647
ActionWizardPresetFlag .....	2647
FRCursorTypeFlags.....	2648
FRDocTypes .....	2648
FRMenuEnableNames .....	2648
FR_RotationFlags .....	2648
EmbedCharSet .....	2649
SpecificChars .....	2649
VTWordStyle.....	2649
FRFormatToolsButtonIDDefs .....	2649
FRMenuNames .....	2650
FRPanelLocation.....	2650
FRSOURCETYPE .....	2650
BuildInToolName .....	2651
FRToolbarNames .....	2652
TextOperations.....	2652
TextOperationUndoRedoType.....	2652
FPDContentParsingState.....	2653
FPDOSTextExtractingFlags .....	2653
FPDTransparencyAttrFlags .....	2653
FRMS_ACTIVATION_FLAGS.....	2653
FRMS_ERR_CODES .....	2654
FPDACTIONTypeNamed.....	2654
FPDJavaScriptString.....	2654
FPDAnnotationFlagBits.....	2654
FPDBookmarkFontFlags .....	2655
FPDConnectedInfoIdTypes .....	2655
FPDConnectedInfoTrackingTypes .....	2655
FPDConnectedOpenActionURLTypes .....	2655
FPDCreatingFlag .....	2656
FPDDestinationZoomMode .....	2656
FPDDocPermissions .....	2656
FPDDocSaveFlags .....	2656
FPDLoadErrCodes .....	2657
FPDFontColorSpaceFamilies .....	2657
FPDFontFlags .....	2657

FPDFFontPredefinedEncoding .....	2658
FPDFFontTypeIDs .....	2658
FPDFFieldTypes .....	2658
FPDComboBoxFieldsAdditionalFlags .....	2659
FPDFFormFieldFlags .....	2659
FPDFFormFieldTextPosition .....	2659
FPDListBoxFieldsAdditionalFlags .....	2659
FPDRadioButtonAdditionalFlags .....	2660
FPDTextFieldsAdditionalFlags .....	2660
FPDFXEncodingType .....	2660
FPDInterFormColorTypes .....	2660
FPDObjTypes .....	2660
FPDPageObjectTypes .....	2661
FPDPathPointFlags .....	2661
FPDParseContextFlags .....	2661
FPDParseErrCodes .....	2662
FPDSearchFlags .....	2662
FPDSearchingStatus .....	2662
FPDRenderOptBitmasksFlag .....	2662
FPDRenderOptColorModeCodes .....	2663
FPDCharInfoFlag .....	2663
FPDTextDirectionsType .....	2663
FPDTextPageFlags .....	2663
FPDWrapperDocGetWrapperType .....	2664
Callback Groups .....	2664
PIHandshakeData_V0100 .....	2664
PIInitUIProcs .....	2665
PISDKData_V0100 .....	2666
PIHDPltfmData .....	2667
FS_FileRead .....	2667
FS_FileStreamCallbacksRec .....	2668
FS_FileWrite .....	2669
FS_Pause .....	2670
FS_StreamWrite .....	2670
FR_ActionHandlerCallbacksRec .....	2671
FR_AppEventCallbacksRec .....	2672
FR_CaptureCallbacksRec .....	2674
FR_ContentProviderCallbacksRec .....	2676
FR_CryptoCallbacksRec .....	2679
FR_DataCollectionHandlerCallbacksRec .....	2681
FR_DocPropertypageCallbacksRec .....	2682
FR_DRMCryptoCallbacksRec .....	2683
FR_DRMSecurityCallbacksRec .....	2683
FR_EmptyFramWndNotifiesRec .....	2684
FR_ExtraPrintInfoProviderCallbackRec .....	2685
FR_MousePtCallbacksRec .....	2687
FR_OwnerFileTypeHandlerCallbacksRec .....	2689
FR_PageEventCallbacksRec .....	2691
FR_PanelViewCallbacksRec .....	2699
FR_PDFAPPluginHandlerCallbacksRec .....	2701

---

FR_POEventCallbacksRec .....	2701
FR_PreferPageCallbacksRec .....	2704
FR_RibbonRecentFileEventCallbacksRec .....	2705
FR_SecurityCallbacksRec .....	2706
FR_SecurityMethodCallbacksRec .....	2707
FR_SelectionCallbacksRec .....	2708
FR_SignatureHandlerCallbacksRec .....	2711
FR_StatusBarWndExCallbacksRec .....	2712
FR_SubscriptionProviderCallbacksRec .....	2713
FR_UndoRedoCallbacksRec .....	2714
FR_WinMSGCallbacksRec .....	2715
FR_WndProviderCallbacksRec .....	2716
FR_ActionWizardHandlerCallbacksRec .....	2718
FR_CloudLoginProviderCallbacksRec .....	2720
FR_CPDFPluginProviderCallbacksRec .....	2721
FR>EditDrawNotifyCallbacksRec .....	2722
FR>EditNotifyCallbacksRec .....	2722
FR>EditOprNotifyCallbacksRec .....	2724
FR>EditUndoItemCallbacksRec .....	2725
FR>VariableTextProviderCallbacksRec .....	2726
FR>EmptyFrameWndViewEventHandlerCallbacksRec .....	2727
FR>FormatToolCallbacksRec .....	2729
FR>FoxitBrowserEventCallbacksRec .....	2732
FR>HTMLEventCallbacksRec .....	2732
FR>MenuOwnerDrawCallbacksRec .....	2733
FR>PanelEventCallbacksRec .....	2734
FR>PropertyToolCallbacksRec .....	2735
FR>ResourcePropertyNotifyCallbacksRec .....	2736
FR>ResourcePropertyPageCallbacksRec .....	2736
FR>ResourcePropertySourceCallbacksRec .....	2738
FR>RibbonFilePageEventCallbacksRec .....	2739
FR>TabBandAddBtnCallbacksRec .....	2740
FR>CmdMsgEventCallbacksRec .....	2740
FR>ToolCallbacksRec .....	2741
FRMS_CommonEventCallbacksRec .....	2743
FRMS_DecryptionCusAuthCallbacksRec .....	2745
FRMS_DecryptionEventCallbacksRec .....	2745
FRMS_EncryptionEventCallbacksRec .....	2747
FPD_BackgroundDraw .....	2748
FPD_CreatorOptionCallbacksRec .....	2749
FPD_EnumPage .....	2749
FPD_FormNotifyCallbacksRec .....	2750
FPD_OCContextCallBack .....	2751
FPD_ProgressiveEncryptCallbacksRec .....	2752

# Preface

---

The Foxit® Reader Plugin SDK Reference is one of several resources available to help you develop Foxit® Reader plug-ins.

## What's in this guide?

This guide provides detailed descriptions for the APIs that can be used to develop plug-ins for Foxit® Reader and Foxit Phantom®.

## Who should read this guide?

This guide is meant for C/C++ developers who would like to create plug-ins to extend or customize Foxit® Reader and Foxit Phantom®.

## Related documentation

*Developing Plug-ins for Foxit Reader*

# Support Layer

## Object list

The Acrobat Support (AS) layer of the core API provides a variety of utility methods, including platform-independent memory allocation and fixed-point math utilities. In addition, it allows plug-ins to replace low-level file system routines used by Acrobat (including read, write, reopen, remove file, rename file, and other directory operations). This enables Acrobat to be used with other file systems, such as on-line systems. Several AS methods return error codes rather than raising exceptions on errors. This is because these methods are called at a low level where exception handling would be inconvenient and expensive.

## Object List

### General

#### FS\_AffineMatrix

#### FS\_Base64Decoder

It represents a Base64 decoder object. You can use it to decode Base64 encoded data. See [FSBase64DecoderNew](#) , [FSBase64DecoderDecode](#) .

#### FS\_Base64Encoder

It represents a Base64 encoder object. You can use it to encode a byte data array into Base64. See [FSBase64EncoderNew](#) , [FSBase64EncoderEncode](#) .

#### FS\_BinaryBuf

Dynamic binary buffers designed for more efficient appending. See [FSBinaryBufNew](#) , [FSBinaryBufDestroy](#) .

#### FS\_ByteArray

A byte array. See [FSByteArrayNew](#) , [FSByteArrayDestroy](#) .

#### FS\_ByteString

An object preparing a byte buffer.

#### FS\_ByteStringArray

A byte string array. See [FSByteStringArrayNew](#) , [FSByteStringArrayDestroy](#) .

#### FS\_CharMap

A map struct for character mappings (encodings).

## **FS\_CodeTransformation**

Code transformation for name, string and text. See [FSCodeTransformationNameDecode](#) , [FSCodeTransformationNameDecode2](#) , [FSCodeTransformationNameEncode](#) , [FSCodeTransformationEncodeString](#) , [FSCodeTransformationDecodeText](#) , [FSCodeTransformationDecodeText2](#) , [FSCodeTransformationEncodeText](#) .

## **FS\_DIBAttribute**

An object representing a device-independent bitmap attribute. The bitmap has the attributes of horizontal resolution, vertical resolution, resolution unit and so on. See [FSDIBitmapLoadInfo](#) .

## **FS\_DIBitmap**

An object representing a device-independent bitmap.

## **FS\_DWordArray**

A double word array. See [FSDWordArrayNew](#) , [FSDWordArrayDestroy](#) .

## **FS\_ExtensionHFTMgr**

It is used to manage the extension HFTs. See [FSExtensionHFTMgrNewHFT](#) , [FSExtensionHFTMgrAddHFT](#) , [FSExtensionHFTMgrGetHFT](#) , [FSExtensionHFTMgrReplaceEntry](#) , [FSExtensionHFTMgrGetEntry](#) .

## **FS\_FileReadHandler**

File reading interface.

## **FS\_FileStream**

The [FS\\_FileStream](#) object is used to read and write stream. See [FSFileStreamNew](#) .

## **FS\_FileWriteHandler**

file writing interface. See [FSFileWriteHandlerNew](#) , [FSFileWriteHandlerDestroy](#) .

## **FS\_FloatRectArray**

A FS\_FloatRect array. See [FSFloatRectArrayNew](#) , [FSFloatRectArrayDestroy](#) .

## **FS\_GUID**

## **FS\_HFT**

An object that describes a set of exported functions. It is an array of function pointers. See [FSExtensionHFTMgrNewHFT](#) , [FSExtensionHFTMgrGetHFT](#) , [FSExtensionHFTMgrReplaceEntry](#) .

## **FS\_MapByteStringToPtr**

FS\_ByteString TO POINTER MAP. See [FSMapByteStringToPtrNew](#) , [FSMapByteStringToPtrDestroy](#) .

## **FS\_MapPtrToPtr**

POINTER/DWORD TO POINTER/DWORD MAP. See [FSMapPtrToPtrNew](#) , [FSMapPtrToPtrDestroy](#) .

## **FS\_PauseHandler**

An object preparing a simple pause instance.

## **FS\_PtrArray**

A typeless pointer array. See [FSPtrArrayNew](#) , [FSPtrArrayDestroy](#) .

## **FS\_StreamWriteHandler**

stream writing interface.

## **FS\_UTF8Decoder**

A UTF8 decoder object. You can use it to decode UTF-8 encoded data. See [FSUTF8DecoderNew](#) , [FSUTF8DecoderDestroy](#) .

## **FS\_UTF8Encoder**

A UTF8 encoder object. You can use it to encode data into UTF-8. See [FSUTF8EncoderNew](#) , [FSUTF8EncoderDestroy](#) .

## **FS\_UUID**

## **FS\_WideString**

An object used to store unicode characters.

## **FS\_WideStringArray**

A wide string array. See [FSWideStringArrayNew](#) , [FSWideStringArrayDestroy](#) .

## FS\_WordArray

A word array. See [FSWordArrayNew](#) , [FSWordArrayDestroy](#) .

## FS\_XMLElement

An object representing a XML element.

# General

## Description

## Definitions

### Definitions summary

#### ERR\_ADDHFT\_EMPTYLHFT

The HFT is an empty object.

#### ERR\_ADDHFT\_NAMEEXIST

Name with the HFT has been used.

#### ERR\_ADDHFT\_SUCCESS

Success.

#### ERR\_ADDHFT\_UNKNOWN

Unknown error.

#### CALL\_REPLACE\_PRO

#### FALSE

Keyword which value is 0.

#### TRUE

Keyword which value is 1.

#### FS\_DEFINEHANDL

Macro to define a handle type.

#### FS\_FORMAT\_CAPITAL

For formating integer: using with FS\_FORMAT\_HEX to produce capital hexadecimal letters

#### FS\_FORMAT\_HEX

For formating integer: using hexadecimal format

#### FS\_FORMAT\_SIGNED

For formating integer: the value is signed.

#### NULL

The null-pointer value.

#### FS\_UIDTYPE\_INVALID

The UUID is invalid.

#### FS\_UIDTYPE\_WINDOWS\_OTHER

The other type.

#### FS\_UIDTYPE\_WINDOWS\_RANDOM

The UUID is constructed by random number only.

#### FS\_UIDTYPE\_WINDOWS\_RANDOM\_HIGHQUALITY

The UUID is constructed by high quality number.

#### FS\_UIDTYPE\_WINDOWS\_TIME\_MAC

The UUID is constructed by time and MAC address.

**FS\_UIDTYPE\_WINDOWS\_TIME\_RANDOM**

The UUID is constructed by time and random number.

**HANDSHAKE\_V0100**

**INIT\_CALLBACK\_STRUC**

**SDK VERSION**

The current SDK version is 9.5.0.0. Built in Foxit Reader/PhantomPDF version 9.5.

## Definitions detail

### ERR\_ADDHFT\_EMPTYLHFT

**Syntax**

```
#define ERR_ADDHFT_EMPTYLHFT 3
```

**Description**

The HFT is an empty object.

**Group**

[AddHFTErrs](#)

**Head file reference**

fs\_common.h: 118

### ERR\_ADDHFT\_NAMEEXIST

**Syntax**

```
#define ERR_ADDHFT_NAMEEXIST 2
```

**Description**

Name with the HFT has been used.

**Group**

[AddHFTErrs](#)

**Head file reference**

fs\_common.h: 115

### ERR\_ADDHFT\_SUCCESS

**Syntax**

```
#define ERR_ADDHFT_SUCCESS 1
```

**Description**

Success.

**Group**

[AddHFTErrs](#)

**Head file reference**

fs\_common.h: 112

**ERR\_ADDHFT\_UNKNOWN****Syntax**

#define ERR\_ADDHFT\_UNKNOWN 0

**Description**

Unknown error.

**Group**[AddHFTErrs](#)**Head file reference**

fs\_common.h: 109

**CALL\_REPLACE\_PRO****Syntax**#define CALL\_REPLACE\_PRO  
(\*((sel##PROTO )(\_gpCoreHFTMgr->GetReplacedEntry(hftSEL, sel, oldProc))))**Description**Calls the previous implementation of a replaced method(that is, the code that would have been executed before the method was replaced using [FSExtensionHFTMgrReplaceEntry](#) ).**Group**[CALL\\_REPLACE\\_PROC](#)**Head file reference**

fs\_common.h: 166

**FALSE****Syntax**

#define FALSE 0

**Description**

Keyword which value is 0.

**Group**[FSBoolean](#)**Head file reference**

fs\_basicExpT.h: 481

## TRUE

**Syntax**

```
#define TRUE 1
```

**Description**

Keyword which value is 1.

**Group**

[FSBoolean](#)

**Head file reference**

fs\_basicExpT.h: 476

## FS\_DEFINEHANDL

**Syntax**

```
#define FS_DEFINEHANDL typedef struct _##name {FS_LPVOID pData;} * name;
```

**Description**

Macro to define a handle type.

**Group**

[FSBoolean](#)

**Head file reference**

fs\_basicExpT.h: 966

## FS\_FORMAT\_CAPITAL

**Syntax**

```
#define FS_FORMAT_CAPITAL 4
```

**Description**

For formating integer: using with FS\_FORMAT\_HEX to produce capital hexadecimal letters

**Group**

[FSFormatingFlags](#)

**Head file reference**

fs\_stringExpT.h: 86

## FS\_FORMAT\_HEX

**Syntax**

```
#define FS_FORMAT_HEX 2
```

**Description**

For formating integer: using hexadecimal format

**Group**

[FSFormattingFlags](#)

**Head file reference**

fs\_stringExpT.h: 83

## FS\_FORMAT\_SIGNED

**Syntax**

```
#define FS_FORMAT_SIGNED 1
```

**Description**

For formating integer: the value is signed.

**Group**

[FSFormattingFlags](#)

**Head file reference**

fs\_stringExpT.h: 80

## NULL

**Syntax**

```
#define NULL 0
```

**Description**

The null-pointer value.

**Group**

[FSNull](#)

**Head file reference**

fs\_basicExpT.h: 497

## FS\_UUIDTYPE\_INVALID

**Syntax**

```
#define FS_UUIDTYPE_INVALID -1
```

**Description**

The UUID is invalid.

**Group**  
[FSUUIDType](#)

**Head file reference**  
fs\_basicExpT.h: 536

## FS\_UUIDTYPE\_WINDOWS\_OTHER

**Syntax**  
`#define FS_UUIDTYPE_WINDOWS_OTHER 0x10001111`

**Description**  
The other type.

**Group**  
[FSUUIDType](#)

**Head file reference**  
fs\_basicExpT.h: 551

## FS\_UUIDTYPE\_WINDOWS\_RANDOM

**Syntax**  
`#define FS_UUIDTYPE_WINDOWS_RANDOM 0x10000010`

**Description**  
The UUID is constructed by random number only.

**Group**  
[FSUUIDType](#)

**Head file reference**  
fs\_basicExpT.h: 548

## FS\_UUIDTYPE\_WINDOWS\_RANDOM\_HIGHQUALITY

**Syntax**  
`#define FS_UUIDTYPE_WINDOWS_RANDOM_HIGHQUALITY 0x10001000`

**Description**  
The UUID is constructed by high quality number.

**Group**  
[FSUUIDType](#)

**Head file reference**

fs\_basicExpT.h: 545

## FS\_UUIDTYPE\_WINDOWS\_TIME\_MAC

### Syntax

```
#define FS_UUIDTYPE_WINDOWS_TIME_MAC 0x10001010
```

### Description

The UUID is constructed by time and MAC address.

### Group

[FSUUIDType](#)

### Head file reference

fs\_basicExpT.h: 539

## FS\_UUIDTYPE\_WINDOWS\_TIME\_RANDOM

### Syntax

```
#define FS_UUIDTYPE_WINDOWS_TIME_RANDOM 0x10001101
```

### Description

The UUID is constructed by time and random number.

### Group

[FSUUIDType](#)

### Head file reference

fs\_basicExpT.h: 542

## HANDSHAKE\_V0100

### Syntax

```
#define HANDSHAKE_V0100 ((1L <
```

### Description

### Group

[Handshakeversion](#)

### Head file reference

fs\_pidata.h: 279

## INIT\_CALLBACK\_STRUC

### Syntax

```
#define INIT_CALLBACK_STRUC (memset(addressOfStruct, 0, sizeOfStruct))
```

**Description**

Init a structure, calling this method to init all callback structure which register into Foxit Reader.

**Group**

[INIT\\_CALLBACK\\_STRUCT](#)

**Head file reference**

fs\_common.h: 184

## SDK\_VERSION

**Syntax**

```
#define SDK_VERSION ((9L<<<
```

**Description**

The current SDK version is 9.5.0.0. Built in Foxit Reader/PhantomPDF version 9.5.

**Group**

[SDKVersion](#)

**Head file reference**

fs\_common.h: 32

## TypeDefs

### TypeDefs summary

**[FS\\_BOOL](#)**

Boolean variable (should be [TRUE](#) or [FALSE](#) ).

**[FS\\_BYTE](#)**

Byte (8 bits).

**[FS\\_CHAR](#)**

8-bit Windows (ANSI) character.

**[FS\\_DWORD](#)**

32-bit unsigned integer.

**[FS\\_FLOAT](#)**

32-bit floating-point number.

**[FS\\_INT32](#)**

32-bit signed integer.

**[FS\\_INT64](#)**

Signed 64-bit integer.

**[FS\\_INTPTR](#)**

Signed integral type for pointer precision.

**[FS\\_LPBYTE](#)**

Pointer to a FS\_BYTE.

**FS\_LPCBYTE**

Pointer to a constant [FS\\_BYTE](#) .

**FS\_LPCSTR**

Pointer to constant 8-bit Windows (ANSI) characters.

**FS\_LPCVOID**

Pointer to any constant type.

**FS\_LPCWSTR**

Pointer to 16-bit unicode characters.

**FS\_LPWORD**

Pointer to a DWORD.

**FS\_LPSTR**

Pointer to 8-bit Windows (ANSI) characters.

**FS\_LPVOID**

Pointer to any type.

**FS\_LPWORD**

Pointer to a [FS\\_WORD](#) .

**FS\_LPWSTR**

Pointer to 16-bit unicode character.

**FS\_POSITION**

A value used to denote the position of an element in a collection.

**FS\_RESULT**

The result code for functoins.

**FS\_SHORT**

Short integer (16 bits).

**FS\_StrSize**

String size is limited to 2^31-1.

**FS\_UINT**

32-bit unsigned integer.

**FS\_UINT64**

Unsigned 64-bit integer.

**FS\_UINTPTR**

Unsigned integral type for pointer precision.

**FS\_UUID[16]**

Type definition of UUID.

**FS\_WCHAR**

16-bit unicode character.

**FS\_WORD**

16-bit unsigned integer.

## Typedefs detail

### **FS\_BOOL**

**Syntax**

```
typedef int FS_BOOL;
```

**Description**

Boolean variable (should be [TRUE](#) or [FALSE](#) ).

**Head file reference**

fs\_basicExpT.h: 391

## FS\_BYTE

### Syntax

```
typedef unsigned char FS_BYTE;
```

### Description

Byte (8 bits).

### Head file reference

fs\_basicExpT.h: 353

## FS\_CHAR

### Syntax

```
typedef char FS_CHAR;
```

### Description

8-bit Windows (ANSI) character.

### Head file reference

fs\_basicExpT.h: 394

## FS\_DWORD

### Syntax

```
#ifdef  
    typedef unsigned int FS_DWORD;  
#else  
    typedef unsigned long FS_DWORD;  
#endif
```

### Description

32-bit unsigned integer.

### Head file reference

fs\_basicExpT.h: 378

## FS\_FLOAT

### Syntax

```
typedef float FS_FLOAT;
```

### Description

32-bit floating-point number.

### Head file reference

fs\_basicExpT.h: 457

## FS\_INT32

### Syntax

```
typedef int FS_INT32;
```

### Description

32-bit signed integer.

### Head file reference

fs\_basicExpT.h: 367

## FS\_INT64

### Syntax

```
#ifdef _MSC_VER  
    typedef __int64 FS_INT64;  
#else  
    typedef long long int FS_INT64;  
#endif
```

### Description

Signed 64-bit integer.

### Head file reference

fs\_basicExpT.h: 431

## FS\_INTPTR

### Syntax

```
#ifdef  
    typedef INT_PTR FS_INTPTR;  
#else  
    typedef int FS_INTPTR;  
#endif
```

### Description

Signed integral type for pointer precision.

### Head file reference

fs\_basicExpT.h: 441

## FS\_LPBYTE

### Syntax

```
typedef unsigned char* FS_LPBYTE;
```

### Description

Pointer to a FS\_BYTE.

**Head file reference**

fs\_basicExpT.h: 355

**FS\_LPCBYTE****Syntax**

```
typedef unsigned char const* FS_LPCBYTE;
```

**Description**Pointer to a constant [FS\\_BYTE](#).**Head file reference**

fs\_basicExpT.h: 358

**FS\_LPCSTR****Syntax**

```
typedef char const* FS_LPCSTR;
```

**Description**

Pointer to constant 8-bit Windows (ANSI) characters.

**Head file reference**

fs\_basicExpT.h: 399

**FS\_LPCVOID****Syntax**

```
typedef void const* FS_LPCVOID;
```

**Description**

Pointer to any constant type.

**Head file reference**

fs\_basicExpT.h: 350

**FS\_LPCWSTR****Syntax**

```
#ifdef  
    typedef unsigned short const* FS_LPCWSTR;  
#else  
    typedef wchar_t const* FS_LPCWSTR;  
#endif
```

**Description**

Pointer to 16-bit unicode characters.

**Head file reference**

fs\_basicExpT.h: 413

**FS\_LPDWORD****Syntax**

```
#ifdef  
    typedef unsigned int* FS_LPDWORD;  
#else  
    typedef unsigned long* FS_LPDWORD;  
#endif
```

**Description**

Pointer to a DWORD.

**Head file reference**

fs\_basicExpT.h: 380

**FS\_LPSTR****Syntax**

```
typedef char* FS_LPSTR;
```

**Description**

Pointer to 8-bit Windows (ANSI) characters.

**Head file reference**

fs\_basicExpT.h: 396

**FS\_LPVOID****Syntax**

```
typedef void* FS_LPVOID;
```

**Description**

Pointer to any type.

**Head file reference**

fs\_basicExpT.h: 347

**FS\_LPWORD****Syntax**

```
typedef unsigned short* FS_LPWORD;
```

**Description**Pointer to a [FS\\_WORD](#).

**Head file reference**

fs\_basicExpT.h: 365

**FS\_LPWSTR****Syntax**

```
#ifdef  
    typedef unsigned short* FS_LPWSTR;  
#else  
    typedef wchar_t* FS_LPWSTR;  
#endif
```

**Description**

Pointer to 16-bit unicode character.

**Head file reference**

fs\_basicExpT.h: 410

**FS\_POSITION****Syntax**

```
typedef void* FS_POSITION;
```

**Description**

A value used to denote the position of an element in a collection.

**Head file reference**

fs\_basicExpT.h: 426

**FS\_RESULT****Syntax**

```
typedef int FS_RESULT;
```

**Description**

The result code for functoins.

**Head file reference**

fs\_basicExpT.h: 460

**FS\_SHORT****Syntax**

```
typedef short FS_SHORT;
```

**Description**

Short integer (16 bits).

**Head file reference**

fs\_basicExpT.h: 361

**FS\_StrSize****Syntax**

```
typedef int FS_StrSize;
```

**Description**

String size is limited to 2^31-1.

**Head file reference**

fs\_stringExpT.h: 68

**FS\_UINT****Syntax**

```
typedef unsigned int FS_UINT;
```

**Description**

32-bit unsigned integer.

**Head file reference**

fs\_basicExpT.h: 402

**FS\_UINT64****Syntax**

```
#ifdef  
    typedef unsigned __int64 FS_UINT64;  
#else  
    typedef unsigned long long FS_UINT64;  
#endif
```

**Description**

Unsigned 64-bit integer.

**Head file reference**

fs\_basicExpT.h: 433

**FS\_UINTPTR****Syntax**

```
#ifdef  
    typedef UINT_PTR FS_UINTPTR;  
#else  
    typedef unsigned int FS_UINTPTR;  
#endif
```

**Description**

Unsigned integral type for pointer precision.

**Head file reference**

fs\_basicExpT.h: 443

**FS\_UUID[16]****Syntax**

```
typedef unsigned char FS_UUID[16];
```

**Description**

Type definition of UUID.

**Head file reference**

fs\_basicExpT.h: 527

**FS\_WCHAR****Syntax**

```
#ifdef _NATIVE_WCHAR_T_DEFINED  
    typedef unsigned short FS_WCHAR;  
#else  
    typedef wchar_t FS_WCHAR;  
#endif
```

**Description**

16-bit unicode character.

**Head file reference**

fs\_basicExpT.h: 407

**FS\_WORD****Syntax**

```
typedef unsigned short FS_WORD;
```

**Description**

16-bit unsigned integer.

**Head file reference**

fs\_basicExpT.h: 363

## Callbacks

### Callbacks summary

**FS\_CALLBACK\_FREEDATA**

Prototype of callback function provided by custom module for deallocating private data.

**FSCharmapGetByteString**

A pointer type to GetByteString function.

**FSCharmapGetWideString**

A pointer type to GetWideString function.

**PIExportHFTsProcType**

It is called by foxit reader to notify the plug-in to export the HFTs.

**PIHandshakeProcType**

It is called by foxit reader to do the handshake.

**PImportReplaceAndRegisterProcType**

It is called by foxit reader to notify the plug-in to import the HFTs.

**PIInitDataProcType**

It is called by foxit reader to notify the plug-in to initialize the data only. The plug-in must initialize the UI in PIInitUIProcs.

**PIMainProcType**

A callback called by Foxit Reader at the first time a plug-in is loading.

**PIMakeTokenProcType**

It is called by foxit reader to validate an internal plug-in.

**PIOAboutPluginsProcType**

A callback called by Foxit Reader to show info of all plug-ins.

**PIOOnDelayLoadJSPluginsProcType**

This callback is invoked by framework to notify the plug-in platform that it is the appropriate time to delay loading js plug-ins.

**PISetupSDKProcType**

It is called by foxit reader to set up the SDK.

**PIUnloadProcType**

It is called by foxit reader to notify the plug-in to unload.

**PIHDGetHandshakeVersion**

Gets the handshakeVersion from Foxit Reader Application.

**PIHDGetAppName**

Gets the application name.

**PIHDRegisterPlugin**

Register a name and a title for plugin to Reader.

**PIHDSetExportHFTsCallback**

Sets the export process.

**PIHDSetImportReplaceAndRegisterCallback**

Sets the replacing process.

**PIHDSetInitDataCallback**

Sets the callback to initialize the data.

**PIHDSetInitUICallbacks**

Sets the callbacks to initialize the UI. If you do not set the callbacks, the plug-in will not be loaded.

**PIHDSetUnloadCallback**

Sets the unload process.

**PILoadMenuBarUI**

It will be invoked when Foxit Reader starts to initialize and load the menu bar.

**PIReleaseMenuBarUI**

It will be invoked to notify the plug-in to release the menu bar.

**PILoadToolBarUI**

It will be invoked when Foxit Reader starts to initialize and load the tool bar.

**PIReleaseToolBarUI**

It will be invoked to notify the plug-in to release the tool bar.

**PILoadRibbonUI**

It will be invoked when Foxit Reader starts to initialize and load the ribbon mode UI.

**PILoadStatusBarUI**

It will be invoked when Foxit Reader starts to initialize and load the status bar.

**PIReleaseStatusBarUI**

It will be invoked to notify the plug-in to release the status bar.

**PIReleaseRibbonUI**

It will be invoked to notify the plug-in to release the ribbon bar.

**PISDGetHandshakeVersion**

Gets the version of handshake structure.

**PISDSetHandshakeProc**

Sets handshake process to Reader.

**PISDGetCoreHFT**

Gets the core HFT manager.

**PISDGetPID**

Gets the Plug-in ID generated by Reader.

**PISetSDKVersion**

Pass the Plug-in SDK version used to Reader. Then Reader will not load the Plug-in whose version is larger.

**PISDSetMakeTokenProc**

Sets make-token process to Reader.

**PIHDLoadPlugin**

Notify Reader to load plug-in(s).

**PIHDSetOnAboutPluginsCallback**

Sets the on about plug-ins process.

**PIHDGetLastPluginLoadError**

Gets the error while loading the last plug-in from Foxit Reader Application.

**PIHDIsPluginDisabledBy**

Tests a plug-in is disabled by some strategy(like GPO) or not.

**PIHDSetOnDelayLoadJSPluginsCallback**

This callback is invoked by framework to notify the plug-in platform that it is the appropriate time to delay loading js plug-ins.

**PIHDLoadPluginUI**

Notify Reader to load plug-in(s) UI.

## Callbacks detail

### FS\_CALLBACK\_FREEDATA

**Syntax**

```
typedef void (*FS_CALLBACK_FREEDATA)(  
    FS_LVOID pData  
)
```

**Description**

Prototype of callback function provided by custom module for deallocating private data.

**Parameter**

---

pData	[In] The data that need to free.
-------	----------------------------------

---

**Return**

void

**Head file reference**

fs\_basicExpT.h: 1122

**Related method**

[FRAnnotSetPrivateData](#)

**FSCharmapGetByteString****Syntax**

```
typedef void (*FSCharmapGetByteString)(  
    FS\_CharMap mapper,  
    FS\_WideString wstr,  
    FS\_ByteString* outBstr  
)
```

**Description**

A pointer type to GetByteString function. The function converts a wide string to a byte string according a character mapper.

**Parameter**

---

mapper	[In] The char mapper for converting.
--------	--------------------------------------

---

---

wstr	[In] A wide string to convert.
------	--------------------------------

---

---

outBstr	[Out] The result string.
---------	--------------------------

---

**Return**

void

**Head file reference**

fs\_stringExpT.h: 115

**FSCharmapGetWideString****Syntax**

```
typedef void (*FSCharmapGetWideString)(  
    FS\_CharMap mapper,  
    FS\_ByteString bstr,  
    FS\_WideString* outWstr  
)
```

**Description**

A pointer type to GetWideString function. The function converts a byte string to a wide string according a character mapper.

**Parameter**


---

mapper	[In] The char mapper for converting.
bstr	[In] A byte string to convert.
outWstr	[Out] The result string.

---

**Return**

void

**Head file reference**

fs\_stringExpT.h: 101

**PIExportHFTsProcType****Syntax**

```
typedef FS_BOOL (*PIExportHFTsProcType)(void );
```

**Description**

It is called by foxit reader to notify the plug-in to export the HFTs.

**Return**

TRUE for success, otherwise FALSE.

**Head file reference**

fs\_pidata.h: 88

**PIHandshakeProcType****Syntax**

```
typedef FS_BOOL (*PIHandshakeProcType)(
    FS_INT32 handshakeVersion,
    void* handshakeData
);
```

**Description**

It is called by foxit reader to do the handshake.

**Parameter**

---

handshakeVersion	[In] The handshakeVersion.
handshakeData	[In] A pointer to <a href="#">PIHandshakeData_V0100</a> structure, it also used topass to <a href="#">PIHDRegisterPlugin()</a> , <a href="#">PIHDSetExportHFTsCallback()</a> , <a href="#">PIHDSetImportReplaceAndRegisterCallback()</a> , <a href="#">()</a> and <a href="#">PIHDSetUnloadCallback()</a> as <i>thisData</i> parameter.

---

**Return**

TRUE for success, otherwise FALSE.

**Head file reference**

fs\_pidata.h: 79

**PIImportReplaceAndRegisterProcType****Syntax**

```
typedef FS_BOOL (*PIImportReplaceAndRegisterProcType)(void );
```

**Description**

It is called by foxit reader to notify the plug-in to import the HFTs.

**Return**

TRUE for success, otherwise FALSE.

**Head file reference**

fs\_pidata.h: 97

**PIInitDataProcType****Syntax**

```
typedef FS_BOOL (*PIInitDataProcType)(void );
```

**Description**

It is called by foxit reader to notify the plug-in to initialize the data only. The plug-in must initialize the UI in PIInitUIProcs.

**Return**

TRUE for success, otherwise FALSE.

**Head file reference**

fs\_pidata.h: 106

**PIMainProcType****Syntax**

---

```
typedef FS_BOOL (*PIMainProcType)(  
    FS_INT32* handshakeVersion,  
    PISetupSDKProcType* setupProc  
);
```

**Description**

A callback called by Foxit Reader at the first time a plug-in is loading.

**Parameter**


---

handshakeVersion	[In] Version of the handshake struct, send by Foxit Reader.
------------------	---

---

setupProc	[Out] A pointer to the handshake procedure.
-----------	---

---

**Return**

If [TRUE](#) , the plug-in will be load continue on. [FALSE](#) , the plug-in will be unload.

**Head file reference**

fs\_pidata.h: 239

**PIMakeTokenProcType****Syntax**

```
typedef void (*PIMakeTokenProcType)(  
    FS_LPCWSTR lpwsEncryptedToken,  
    void* bufSourceToken,  
    FS_INT32 nBufLength  
);
```

**Description**

It is called by foxit reader to validate an internal plug-in.

**Parameter**


---

lpwsEncryptedToken	[In] The encrypted token.
--------------------	---------------------------

---

bufSourceToken	[Out] Buffer for output source token.
----------------	---------------------------------------

---

nBufLength	[In] Buffer length.
------------	---------------------

---

**Return**

void.

**Head file reference**

fs\_pidata.h: 66

## PIOOnAboutPluginsProcType

### Syntax

```
typedef void (*PIOOnAboutPluginsProcType)(  
    void* pluginData  
)
```

### Description

A callback called by Foxit Reader to show info of all plug-ins.

### Parameter

---

pluginData	[In] Data of loaded plug-ins.
------------	-------------------------------

---

### Return

#### Head file reference

fs\_pidata.h: 250

## PIOOnDelayLoadJSPluginsProcType

### Syntax

```
typedef void (*PIOOnDelayLoadJSPluginsProcType)(  
    FS_BOOL bSilent,  
    void* pParentWnd  
)
```

### Description

This callback is invoked by framework to notify the plug-in platform that it is the appropriate time to delay loading js plug-ins.

### Parameter

---

bSilent	[In] It indicates whether the js plug-in is loaded silently or not.
---------	---

---

pParentWnd	[In] A pointer to the parent window. You can convert it to <i>MFC CWnd*</i> .
------------	---

---

### Return

#### Head file reference

fs\_pidata.h: 262

## PISetupSDKProcType

**Syntax**

```
typedef FS_BOOL (*PISetupSDKProcType)(  
    FS_INT32 handshakeVersion,  
    void* sdkData  
>;
```

**Description**

It is called by foxit reader to set up the SDK.

**Parameter**

handshakeVersion	[In] The version of the handshake.
sdkData	[In] A pointer to <a href="#">PISDKData_V0100</a> structure, it also be used to pass to <a href="#">PISDSetHandshakeProc</a> () as <i>thisData</i> parameter.

**Return**

TRUE for success, otherwise FALSE.

**Head file reference**

fs\_pidata.h: 54

**PIUnloadProcType****Syntax**

```
typedef FS_BOOL (*PIUnloadProcType)(void );
```

**Description**

It is called by foxit reader to notify the plug-in to unload.

**Return**

TRUE for success, otherwise FALSE.

**Head file reference**

fs\_pidata.h: 226

**PIHDGetHandshakeVersion****Syntax**

```
typedef FS_INT32 (*PIHDGetHandshakeVersion)(void );
```

**Description**

Gets the handshakeVersion from Foxit Reader Application.

**Return**

The build-in handshakeVersion of Foxit Reader Application.

**Head file reference**

fs\_pidata.h: 455

**Group**[PIHandshakeData\\_V0100](#)**PIHDGetAppName****Syntax**

```
typedef char* (*PIHDGetAppName)(void );
```

**Description**

Gets the application name.

**Return**

The application name.

**Head file reference**

fs\_pidata.h: 464

**Group**[PIHandshakeData\\_V0100](#)**PIHDRegisterPlugin****Syntax**

```
typedef FS_BOOL (*PIHDRegisterPlugin)(  
    void* thisData,  
    char* name,  
    FS\_LPCWSTR lpwsTitle  
)
```

**Description**

Register a name and a title for plugin to Reader.

**Parameter**

---

thisData	[In] A pointer to a PIHandshakeData_V0100 structure which passed by <a href="#">PIHandshakeProcType</a> ().
----------	---

---

name	[In] The plug-in name.
------	------------------------

---

lpwsTitle	[In] The title to be displayed.
-----------	---------------------------------

---

**Return**

TRUE means successful, otherwise not.

#### **Head file reference**

fs\_pidata.h: 476

#### **Group**

[PIHandshakeData\\_V0100](#)

### PIHDSetExportHFTsCallback

#### **Syntax**

```
typedef void (*PIHDSetExportHFTsCallback)(
    void* thisData,
    PIExportHFTsProcType proc
);
```

#### **Description**

Sets the export process.

#### **Parameter**

---

thisData	[In] A pointer to a PIHandshakeData_V0100 structure which passed by <a href="#">PIHandshakeProcType</a> ().
----------	---

---

proc	[In] A user-supplied callback to call whenever plug-ins need to export its owner HFT.
------	---

---

#### **Return**

#### **Head file reference**

fs\_pidata.h: 487

#### **Group**

[PIHandshakeData\\_V0100](#)

### PIHDSetImportReplaceAndRegisterCallback

#### **Syntax**

```
typedef void (*PIHDSetImportReplaceAndRegisterCallback)(
    void* thisData,
    PIImportReplaceAndRegisterProcType proc
);
```

#### **Description**

Sets the replacing process.

#### **Parameter**

---

thisData	[In] A pointer to a PIHandshakeData_V0100 structure which passed by <a href="#">PIHandshakeProcType</a> ().
----------	---

---

proc	[In] A user-supplied callback to call whenever plug-ins need to replace or register some entry.
------	---

---

**Return****Head file reference**

fs\_pidata.h: 498

**Group**[PIHandshakeData\\_V0100](#)**PIHDSetInitDataCallback****Syntax**

```
typedef void (*PIHDSetInitDataCallback)(
    void* thisData,
    PIInitDataProcType proc
);
```

**Description**

Sets the callback to initialize the data.

**Parameter**


---

thisData	[In] A pointer to a PIHandshakeData_V0100 structure which passed by <a href="#">PIHandshakeProcType</a> ().
----------	---

---

proc	[In] A user-supplied callback to call whenever plug-ins need to do some initialization.
------	---

---

**Return****Head file reference**

fs\_pidata.h: 511

**Group**[PIHandshakeData\\_V0100](#)**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)**PIHDSetInitUICallbacks**

**Syntax**

```
typedef void (*PIHDSetInitUICallbacks)(  
    void* thisData,  
    PIInitUIProcs* pProcs  
)
```

**Description**

Sets the callbacks to initialize the UI. If you do not set the callbacks, the plug-in will not be loaded.

**Parameter**

---

thisData	[In] A pointer to a PIHandshakeData_V0100 structure which passed by <a href="#">PIHandshakeProcType</a> ().
----------	---

---

pProcs	[In] The user-supplied callbacks to call when plug-ins need to initialize the UI.
--------	---

---

**Return****Head file reference**

fs\_pidata.h: 524

**Group**

[PIHandshakeData\\_V0100](#)

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**PIHDSetUnloadCallback****Syntax**

```
typedef void (*PIHDSetUnloadCallback)(  
    void* thisData,  
    PIUnloadProcType proc  
)
```

**Description**

Sets the unload process.

**Parameter**

---

thisData	[In] A pointer to a PIHandshakeData_V0100 structure which passed by <a href="#">PIHandshakeProcType</a> ().
----------	---

---

---

proc	[In] A user-supplied callback to call whenever plug-ins need to unload.
------	---

---

**Return****Head file reference**

fs\_pidata.h: 535

**Group**[PIHandshakeData\\_V0100](#)**PILoadMenuBarUI****Syntax**

```
typedef void (*PILoadMenuBarUI)(
    void* pParentWnd
);
```

**Description**

It will be invoked when Foxit Reader starts to initialize and load the menu bar.

**Parameter**


---

pParentWnd	[In] A pointer to the parent window. You can convert it to <i>MFC CWnd</i> *.
------------	---

---

**Return**

void

**Head file reference**

fs\_pidata.h: 128

**Group**[PIInitUIProcs](#)**Related method****Since**[SDK LATEEST VERSION > 2.1.0.4](#)**PIReleaseMenuBarUI****Syntax**

```
typedef void (*PIReleaseMenuBarUI)(
    void* pParentWnd
);
```

**Description**

It will be invoked to notify the plug-in to release the menu bar.

**Parameter**

---

pParentWnd	[In] A pointer to the parent window. You can convert it to <i>MFC CWnd*</i> .
------------	---

---

**Return**

void

**Head file reference**

fs\_pidata.h: 140

**Group**

[PIInitUIProcs](#)

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**PILoadToolBarUI****Syntax**

```
typedef void (*PILoadToolBarUI)(  
    void* pParentWnd  
)
```

**Description**

It will be invoked when Foxit Reader starts to initialize and load the tool bar.

**Parameter**

---

pParentWnd	[In] A pointer to the parent window. You can convert it to <i>MFC CWnd*</i> .
------------	---

---

**Return**

void

**Head file reference**

fs\_pidata.h: 152

**Group**

[PIInitUIProcs](#)

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)**PIReleaseToolBarUI****Syntax**

```
typedef void (*PIReleaseToolBarUI)(  
    void* pParentWnd  
)
```

**Description**

It will be invoked to notify the plug-in to release the tool bar.

**Parameter**

---

pParentWnd	[In] A pointer to the parent window. You can convert it to <i>MFC CWnd*</i> .
------------	---

---

**Return**

void

**Head file reference**

fs\_pidata.h: 164

**Group**

[PIInitUIProcs](#)

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**PILoadRibbonUI****Syntax**

```
typedef void (*PILoadRibbonUI)(  
    void* pParentWnd  
)
```

**Description**

It will be invoked when Foxit Reader starts to initialize and load the ribbon mode UI.

**Parameter**

---

pParentWnd	[In] A pointer to the parent window. You can convert it to <i>MFC CWnd*</i> .
------------	---

---

**Return**

void

**Head file reference**

fs\_pidata.h: 176

**Group**

[PIInitUIProcs](#)

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**PILoadStatusBarUI****Syntax**

```
typedef void (*PILoadStatusBarUI)(  
    void* pParentWnd  
)
```

**Description**

It will be invoked when Foxit Reader starts to initialize and load the status bar.

**Parameter**

---

pParentWnd	[In] A pointer to the parent window. You can convert it to <i>MFC CWnd*</i> .
------------	---

---

**Return**

void

**Head file reference**

fs\_pidata.h: 188

**Group**

[PIInitUIProcs](#)

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**PIReleaseStatusBarUI****Syntax**

```
typedef void (*PIReleaseStatusBarUI)(  
    void* pParentWnd  
)
```

**Description**

It will be invoked to notify the plug-in to release the status bar.

**Parameter**

---

pParentWnd	[In] A pointer to the parent window. You can convert it to <i>MFC CWnd*</i> .
------------	---

---

**Return**

void

**Head file reference**

fs\_pidata.h: 200

**Group**

[PIInitUIProcs](#)

**Related method****Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**PIReleaseRibbonUI****Syntax**

```
typedef void (*PIReleaseRibbonUI)(  
    void* pParentWnd  
)
```

**Description**

It will be invoked to notify the plug-in to release the ribbon bar.

**Parameter**

---

pParentWnd	[In] A pointer to the parent window. You can convert it to <i>MFC CWnd*</i> .
------------	---

---

**Return**

void

**Head file reference**

fs\_pidata.h: 212

**Group**

[PIInitUIProcs](#)

**Related method**

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**PISDGetHandshakeVersion****Syntax**

```
typedef FS_INT32 (*PISDGetHandshakeVersion)(void );
```

**Description**

Gets the version of handshake structure.

**Return**

The handshake version.

**Head file reference**

fs\_pidata.h: 301

**Group**

[PISDKData\\_V0100](#)

**PISDSetHandshakeProc****Syntax**

```
typedef void (*PISDSetHandshakeProc)(  
    void* thisData,  
    PIHandshakeProcType proc  
)
```

**Description**

Sets handshake process to Reader.

**Parameter**

---

thisData	[In] A pointer to a PISDKData_V0100 structure which passed by <a href="#">PISetupSDKProcType</a> ().
----------	--

---

proc	[In] The input handshake callback.
------	------------------------------------

---

**Return****Head file reference**

fs\_pidata.h: 312

**Group**

[PISDKData\\_V0100](#)

## PISDGetCoreHFT

### Syntax

```
typedef FRCoreHFTMgr* (*PISDGetCoreHFT)(void );
```

### Description

Gets the core HFT manager.

### Return

The core HFT manager.

### Head file reference

fs\_pidata.h: 321

### Group

[PISDKData\\_V0100](#)

## PISDGetPID

### Syntax

```
typedef FS_LPVVOID (*PISDGetPID)(  
    void* thisData  
) ;
```

### Description

Gets the Plug-in ID generated by Reader.

### Parameter

---

thisData	[In] A pointer to a PISDKData_V0100 structure which passed by <a href="#">PISetupSDKProcType</a> ().
----------	--

---

### Return

The plug-in ID.

### Head file reference

fs\_pidata.h: 331

### Group

[PISDKData\\_V0100](#)

## PISetSDKVersion

### Syntax

```
typedef void (*PISetSDKVersion)(  
    void* thisData,  
    FS\_INT32 nPISDKVersion  
) ;
```

**Description**

Pass the Plug-in SDK version used to Reader. Then Reader will not load the Plug-in whose version is larger.

**Parameter**

---

thisData	[In] A pointer to a PISDKData_V0100 structure which passed by <a href="#">PISetupSDKProcType</a> ().
nPISDKVersion	[In] The version of the Plug-in SDK.

---

**Return****Head file reference**

fs\_pidata.h: 342

**Group**

[PISDKData\\_V0100](#)

**PISDSetMakeTokenProc****Syntax**

```
typedef void (*PISDSetMakeTokenProc)(  
    void* thisData,  
    PIMakeTokenProcType proc  
)
```

**Description**

Sets make-token process to Reader.

**Parameter**

---

thisData	[In] A pointer to a PISDKData_V0100 structure which passed by <a href="#">PISetupSDKProcType</a> ().
proc	[In] The input make-token callback.

---

**Return****Head file reference**

fs\_pidata.h: 353

**Group**

[PISDKData\\_V0100](#)

**PIHDLoadPlugin****Syntax**

```
typedef FS_BOOL (*PIHDLoadPlugin)(
    void* arrLibPath,
    FS_BOOL bFreeLibWhenFailed
);
```

**Description**

Notify Reader to load plug-in(s).

**Parameter**


---

arrLibPath	[In] A pointer to an array of library path.
bFreeLibWhenFailed	[In] Flag to indicate whether to free library when failed load plug-in(s).

---

**Return**

TRUE means successful, otherwise not.

**Head file reference**

fs\_pidata.h: 377

**Group**

[PIHDPItfmData](#)

**PIHDSetOnAboutPluginsCallback****Syntax**

```
typedef void (*PIHDSetOnAboutPluginsCallback)(
    PIOntAboutPluginsProcType proc
);
```

**Description**

Sets the on about plug-ins process.

**Parameter**


---

proc	[In] A callback supplied by plug-in platform to call to manage plug-ins.
------	--

---

**Return****Head file reference**

fs\_pidata.h: 388

**Group**

[PIHDPltfmData](#)**PIHDGetLastPluginLoadError****Syntax**

```
typedef FS_INT32 (*PIHDGetLastPluginLoadError)(void );
```

**Description**

Gets the error while loading the last plug-in from Foxit Reader Application.

**Return**

The error code while loading the last plug-in.

**Head file reference**

fs\_pidata.h: 397

**Group**[PIHDPltfmData](#)**PIHDIIsPluginDisabledBy****Syntax**

```
typedef FS_BOOL (*PIHDIIsPluginDisabledBy)(  
    FS_LPCWSTR lpwsFileName  
) ;
```

**Description**

Tests a plug-in is disabled by some strategy(like GPO) or not.

**Parameter**

---

lpwsFileName	[In] The file name of the tested plug-in.
--------------	---

---

**Return**

TRUE means yes, otherwise not.

**Head file reference**

fs\_pidata.h: 407

**Group**[PIHDPltfmData](#)**PIHDSetOnDelayLoadJSPluginsCallback****Syntax**

```
typedef void (*PIHDSetOnDelayLoadJSPluginsCallback)(
```

---

```
PIOOnDelayLoadJSPluginsProcType proc
);
```

**Description**

This callback is invoked by framework to notify the plug-in platform that it is the appropriate time to delay loading js plug-ins.

**Parameter**


---

proc	[In] A callback supplied by plug-in platform and invoked by framework.
------	--

---

**Return****Head file reference**

fs\_pidata.h: 418

**Group**

[PIHDPltfmData](#)

**PIHDLoadPluginUI****Syntax**

```
typedef FS_BOOL (*PIHDLoadPluginUI)(
    HWND hWnd,
    void* arrLibPath
);
```

**Description**

Notify Reader to load plug-in(s) UI.

**Parameter**


---

hWnd	[In] The main frame window of Foxit Reader.
------	---

---

arrLibPath	[In] Self-defined data structure.
------------	-----------------------------------

---

**Return**

TRUE means successful, otherwise not.

**Head file reference**

fs\_pidata.h: 429

**Group**

[PIHDPltfmData](#)

## Structures

### Structures summary

#### [FRCoreHFTMgr](#)

Core HFT manager.

#### [FS\\_AffineMatrix](#)

A data structure representing a affine-matrix object.

#### [FS\\_DevicePoint](#)

A data structure representing a point in a window's coordinate space. See [FRAppGetMousePos](#) , [FRDocGetPageViewAtPoint](#) , [FRPageViewGetAnnotAtPoint](#) , [FRPageViewDrawLine](#) , [FRPageViewDrawPolygon](#) , [FRPageViewDrawPolygonOutline](#) .

#### [FS\\_FloatPoint](#)

A float-point coordination point. See [FRPageViewDevicePointToPage](#) , [FRPageViewPointToDevice](#) .

#### [FS\\_FloatRect](#)

A float-point coordination rectangle.

#### [FS\\_GUID](#)

Structure for GUID and type definitions.

#### [FS\\_PATHPOINT](#)

Point struct used in [FPD\\_Path](#) . See [FPDPathGetPoint](#) .

#### [FS\\_Rect](#)

A data structure representing a rectangle (a quadrilateral having only horizontal and vertical sides) in a window's coordinate space.

### Structs detail

#### FRCoreHFTMgr

##### Syntax

```
typedef struct __FRCoreHFTMgr__{
    unsigned long lStructSize,
    void* (*QueryInterface)(int hftSel, int iSel, void* gID),
    void (*UpdateInterface)(int hftSel, int iSel, void* gID),
    int (*FRGetSDKVersion)(void),
    void* (*GetReplacedEntry)(int hftSel, int iSel, void* oldProc)
}FRCoreHFTMgr;
```

##### Description

Core HFT manager.

##### Head file reference

fs\_common.h: 50

##### long lStructSize

The size of data structure. It must be set to *sizeof*( [FRCoreHFTMgr](#) )

**(\*QueryInterface)(int hftSel, int iSel, void\* gID)**

**(\*UpdateInterface)(int hftSel, int iSel, void\* gID)**

**(\*FRGetSDKVersion)(void)**

---

(\*GetReplacedEntry)(int hftSel, int iSel, void\* oldProc)

## FS\_AffineMatrix

### Syntax

```
typedef struct __FS_AffineMatrix__{
    FS_FLOAT a,
    FS_FLOAT b,
    FS_FLOAT c,
    FS_FLOAT d,
    FS_FLOAT e,
    FS_FLOAT f
}FS_AffineMatrix;
```

### Description

A data structure representing a affine-matrix object. It defines six coefficients: a, b, c, d, e, f. The transformation equations are: dest\_x = a \* src\_x + c \* src\_y + e; dest\_y = b \* src\_x + d \* src\_y + f; See [FSAffineMatrixGetReverse](#) , [FSAffineMatrixTransformPoint](#) , [FSAffineMatrixTransformRect](#) , [FSAffineMatrixConcat](#) , [FSAffineMatrixTransformDistance](#) , [FRPageViewGetCurrentMatrix](#) .

### Head file reference

fs\_basicExpT.h: 940

#### a

The coefficient a.

#### b

The coefficient b.

#### c

The coefficient c.

#### d

The coefficient d.

#### e

The coefficient e.

#### f

The coefficient f.

## FS\_DevicePoint

### Syntax

```
typedef struct _FS_DevicePoint_{
    int x,
    int y
}FS_DevicePoint, *PFS_DevicePoint;
```

### Description

A data structure representing a point in a window's coordinate space. See [FRAppGetMousePos](#) , [FRDocGetPageViewAtPoint](#) , [FRPageViewGetAnnotAtPoint](#) , [FRPageViewDrawLine](#) , [FRPageViewDrawPolygon](#) , [FRPageViewDrawPolygonOutline](#) .

#### **Head file reference**

fs\_basicExpT.h: 915

**x**

x-coordinate of the point.

**y**

y-coordinate of the point.

### FS\_FloatPoint

#### **Syntax**

```
typedef struct {
    FS\_FLOAT m_PointX,
    FS\_FLOAT m_PointY
}FS_FloatPoint;
```

#### **Description**

A float-point coordination point. See [FRPageViewDevicePointToPage](#) , [FRPageViewPointToDevice](#) .

#### **Head file reference**

fs\_basicExpT.h: 894

#### **m\_PointX**

x-coordinate of the point.

#### **m\_PointY**

y-coordinate of the point.

### FS\_FloatRect

#### **Syntax**

```
typedef struct __FS_FloatRect__{
    FS\_FLOAT left,
    FS\_FLOAT bottom,
    FS\_FLOAT right,
    FS\_FLOAT top
}FS_FloatRect;
```

#### **Description**

A float-point coordination rectangle. A normalized float-point rectangle always has top larger than bottom, which is the opposite of Windows rectangle.

#### **Head file reference**

fs\_basicExpT.h: 855

**left**

The x-coordinate of the left-bottom corner.

**bottom**

The y-coordinate of the left-bottom corner.

**right**

The x-coordinate of the right-top corner.

**top**

The y-coordinate of the right-top corner.

## FS\_GUID

**Syntax**

```
typedef struct __FS_GUID__{
    FS_DWORD data1,
    FS_WORD data2,
    FS_WORD data3,
    FS_BYTE data4[8]
}FS_GUID, *FS_LPGUID;
```

**Description**

Structure for GUID and type definitions.

**Head file reference**

fs\_basicExpT.h: 511

**data1**

Data1.

**data2**

Data2.

**data3**

Data3.

**data4[8]**

Data4.

## FS\_PATHPOINT

**Syntax**

```
typedef struct {
    FS_FLOAT m_PointX,
    FS_FLOAT m_PointY,
    int m_Flag
}FS_PATHPOINT;
```

**Description**

Point struct used in [FPD\\_Path](#) . See [FPDPathGetPoint](#) .

**Head file reference**

fs\_basicExpT.h: 877

**m\_PointX**

x-coordinate of the point.

**m\_PointY**

y-coordinate of the point.

**m\_Flag**

flags of the point.

**FS\_Rect****Syntax**

```
typedef struct __FS_RECT__{
    int left,
    int top,
    int right,
    int bottom
}FS_Rect;
```

**Description**

A data structure representing a rectangle (a quadrilateral having only horizontal and vertical sides) in a window's coordinate space.

**Head file reference**

fs\_basicExpT.h: 832

**left**

The x-coordinate of the upper-left corner of a rectangle.

**top**

The y-coordinate of the upper-left corner of a rectangle.

**right**

The x-coordinate of the lower-right corner of a rectangle.

**bottom**

The y-coordinate of the lower-right corner of a rectangle.

**FS\_AffineMatrix**[Return from Used by](#)**Description****Returned from**

[FRPageViewGetCurrentMatrix](#)  
[FPDArrayGetMatrix](#)  
[FPDefaultAppearanceGetTextMatrix](#)  
[FPDictionaryGetMatrix](#)  
[FPDInlineImagesGetMatrix](#)  
[FPDPageGetDisplayMatrix](#)  
[FPDPageGetPageMatrix](#)  
[FPDPatternGetPatternMatrix](#)  
[FPDShadingPatternGetPatternMatrix](#)  
[FPDTilingPatternGetPatternMatrix](#)  
[FPDType3CharGetImageMatrix](#)  
[FPDType3FontGetFontMatrix](#)  
[FSAffineMatrixGetReverse](#)  
[FSAffineMatrixConcat](#)  
[FSAffineMatrixTranslate](#)  
[FSAffineMatrixTranslateI](#)  
[FSAffineMatrixScale](#)  
[FSAffineMatrixConcatInverse](#)  
[FSAffineMatrixMatchRect](#)  
[FSAffineMatrixRotate](#)

#### Used by

[FSDIBitmapTransformTo](#)  
[FPDAnnotDrawAppearance](#)  
[FPDAnnotDrawBorder](#)  
[FPDAnnotDrawInContext](#)  
[FPDAnnotListDisplayAnnots](#)  
[FPDAnnotListDisplayAnnotsEx](#)  
[FPDAnnotListGetAnnotMatrix](#)  
[FPDAnnotListGetAnnotRect](#)  
[FPDefaultAppearanceSetTextMatrix](#)  
[FPDDictionarySetAtMatrix](#)  
[FPDFFormParseContent](#)  
[FPDFFormStartParse](#)  
[FPDFFormTransform](#)  
[FPDFFormControlDrawControl](#)  
[FPDFFormObjectGetTransformMatrix](#)  
[FPDFFormObjectTransform](#)  
[FPDImageObjectGetTransformMatrix](#)  
[FPDImageObjectTransform](#)  
[FPDMeshStreamGetVertex](#)  
[FPDMeshStreamGetVertexRow](#)  
[FPDPageTransform](#)  
[FPDPageObjectGetBBox](#)  
[FPDPageObjectTransformClipPath](#)  
[FPDPathAppend](#)  
[FPDPathTransform](#)  
[FPDPathObjectGetTransformMatrix](#)  
[FPDPathObjectTransform](#)  
[FPDRenderContextAppendForm](#)  
[FPDRenderContextAppendPage](#)  
[FPDRenderContextDrawForm](#)  
[FPDRenderContextDrawPage](#)  
[FPDRenderContextDrawStream](#)  
[FPDRenderContextGetBackground](#)

[\*\*FPDRenderContextRender\*\*](#)  
[\*\*FPDRenderDeviceDrawNormalText\*\*](#)  
[\*\*FPDRenderDeviceDrawPath\*\*](#)  
[\*\*FPDRenderDeviceDrawTextPath\*\*](#)  
[\*\*FPDRenderDeviceDrawTextString2\*\*](#)  
[\*\*FPDRenderDeviceDrawType3Text\*\*](#)  
[\*\*FPDRenderDeviceSetClip\\_PathFill\*\*](#)  
[\*\*FPDShadingObjectGetTransformMatrix\*\*](#)  
[\*\*FPDShadingObjectTransform\*\*](#)  
[\*\*FPDTextObjectGetTextMatrix\*\*](#)  
[\*\*FPDTextObjectTransform\*\*](#)  
[\*\*FPDTilingPatternNewII\*\*](#)  
[\*\*FSAffineMatrixIsScaled\*\*](#)  
[\*\*FSAffineMatrixIs90Rotated\*\*](#)  
[\*\*FSAffineMatrixGetReverse\*\*](#)  
[\*\*FSAffineMatrixTransformPoint\*\*](#)  
[\*\*FSAffineMatrixTransformRect\*\*](#)  
[\*\*FSAffineMatrixConcat\*\*](#)  
[\*\*FSAffineMatrixTranslate\*\*](#)  
[\*\*FSAffineMatrixTranslateI\*\*](#)  
[\*\*FSAffineMatrixScale\*\*](#)  
[\*\*FSAffineMatrixConcatInverse\*\*](#)  
[\*\*FSAffineMatrixGetUnitRect\*\*](#)  
[\*\*FSAffineMatrixGetUnitArea\*\*](#)  
[\*\*FSAffineMatrixGetXUnit\*\*](#)  
[\*\*FSAffineMatrixGetYUnit\*\*](#)  
[\*\*FSAffineMatrixTransformDistance\*\*](#)  
[\*\*FSAffineMatrixRotate\*\*](#)

## Functions

### Functions summary

#### [\*\*FSAffineMatrixIsScaled\*\*](#)

Whether this matrix has scaling (or translating) only. No rotating.

#### [\*\*FSAffineMatrixIs90Rotated\*\*](#)

Whether this matrix has rotating of 90, or -90 degrees.

#### [\*\*FSAffineMatrixGetReverse\*\*](#)

Gets the inverse matrix base on a source matrix.

#### [\*\*FSAffineMatrixTransformPoint\*\*](#)

Transforms a point.

#### [\*\*FSAffineMatrixTransformRect\*\*](#)

Transforms a rectangle and return a bounding rectangle.

#### [\*\*FSAffineMatrixConcat\*\*](#)

Concatenates with another matrix.

#### [\*\*FSAffineMatrixTranslate\*\*](#)

Translates the matrix.

#### [\*\*FSAffineMatrixTranslateI\*\*](#)

Translates the matrix using integer value.

#### [\*\*FSAffineMatrixScale\*\*](#)

Scales the matrix.

#### [\*\*FSAffineMatrixConcatInverse\*\*](#)

Concatenates the inverse of another matrix.

**[FSAffineMatrixMatchRect](#)**

Gets a matrix that transforms a source rectangle to dest rectangle.

**[FSAffineMatrixGetUnitRect](#)**

Gets a bounding rectangle of the parallelogram composing two unit vectors.

**[FSAffineMatrixGetUnitArea](#)**

Gets area of the parallelogram composing two unit vectors.

**[FSAffineMatrixGetXUnit](#)**

Gets the x-direction unit size.

**[FSAffineMatrixGetYUnit](#)**

Gets the y-direction unit size.

**[FSAffineMatrixTransformDistance](#)**

Transforms a distance.

**[FSAffineMatrixRotate](#)**

Rotates the matrix.

## Functions detail

**FSAffineMatrixIsScaled****Syntax**

```
FS_BOOL FSAffineMatrixIsScaled (
    FS_AffineMatrix matrix
);
```

**Description**

Whether this matrix has scaling (or translating) only. No rotating.

**Parameter**

matrix	[In] The input matrix.
--------	------------------------

**Return**

[TRUE](#) for this matrix having scaling (or translating) only, no rotating. Otherwise not.

**Head file reference**

fs\_basicTempl.h: 86

**Related method****[FSAffineMatrixScale](#)****FSAffineMatrixIs90Rotated****Syntax**

```
FS_BOOL FSAffineMatrixIs90Rotated (
    FS_AffineMatrix matrix
);
```

**Description**

Whether this matrix has rotating of 90, or -90 degrees.

**Parameter**

---

matrix	[In] The input matrix.
--------	------------------------

---

**Return**

[TRUE](#) for this matrix having rotating, otherwise not.

**Head file reference**

fs\_basicTempl.h: 96

**FSAffineMatrixGetReverse****Syntax**

```
FS_AffineMatrix FSAffineMatrixGetReverse (
    FS\_AffineMatrix src
);
```

**Description**

Gets the inverse matrix base on a source matrix.

**Parameter**

---

src	[In] The input matrix.
-----	------------------------

---

**Return**

The inverse matrix base on a source matrix.

**Head file reference**

fs\_basicTempl.h: 105

**FSAffineMatrixTransformPoint****Syntax**

```
void FSAffineMatrixTransformPoint (
    FS\_AffineMatrix matrix,
    FS\_FLOAT x,
    FS\_FLOAT y,
    FS\_FLOAT* outX,
    FS\_FLOAT* outY
);
```

**Description**

Transforms a point.

**Parameter**

---

matrix	[In] The input matrix.
x	[In] The input x-coordinate of the point
y	[In] The input y-coordinate of the point.
outX	[Out] It receives the output transformed x-coordinate.
outY	[Out] It receives the output transformed y-coordinate.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 114

**Related method**

[FSAffineMatrixTransformRect](#)  
[FSAffineMatrixTransformDistance](#)

**FSAffineMatrixTransformRect****Syntax**

```
FS_FloatRect FSAffineMatrixTransformRect (
    FS_AffineMatrix matrix,
    FS_FloatRect rc
);
```

**Description**

Transforms a rectangle and return a bounding rectangle. The result rectangle is always normalized.

**Parameter**


---

matrix	[In] The input matrix.
rc	[In] Rectangle to transform.

---

**Return**

The result rectangle.

**Head file reference**

fs\_basicTempl.h: 123

**Related method**

[FSAffineMatrixTransformPoint](#)  
[FSAffineMatrixTransformDistance](#)

## FSAffineMatrixConcat

### Syntax

```
FS_AffineMatrix FSAffineMatrixConcat (
    FS_AffineMatrix matrix,
    FS_AffineMatrix src
);
```

### Description

Concatenates with another matrix.

### Parameter

matrix	[In] The input matrix.
src	[In] The input matrix.

### Return

The result matrix.

### Head file reference

fs\_basicTempl.h: 141

### Related method

[FSAffineMatrixConcatInverse](#)

## FSAffineMatrixTranslate

### Syntax

```
FS_AffineMatrix FSAffineMatrixTranslate (
    FS_AffineMatrix matrix,
    FS_FLOAT x,
    FS_FLOAT y
);
```

### Description

Translates the matrix.

### Parameter

matrix	[In] The input matrix.
--------	------------------------

---

x	[In] The x-direction delta value.
---	-----------------------------------

---

y	[In] The y-direction delta value.
---	-----------------------------------

---

**Return**

The result matrix.

**Head file reference**

fs\_basicTempl.h: 152

**Related method**

[FSAffineMatrixTranslateI](#)

[FSAffineMatrixTranslateI](#)

**Syntax**

```
FS_AffineMatrix FSAffineMatrixTranslateI (
    FS\_AffineMatrix matrix,
    FS\_INT32 x,
    FS\_INT32 y
);
```

**Description**

Translates the matrix using integer value.

**Parameter**


---

matrix	[In] The input matrix.
--------	------------------------

---

x	[In] The x-direction delta integer value.
---	---

---

y	[In] The y-direction delta integer value.
---	---

---

**Return**

The result matrix.

**Head file reference**

fs\_basicTempl.h: 159

**Related method**

[FSAffineMatrixTranslate](#)

[FSAffineMatrixScale](#)

**Syntax**

---

```
FS_AffineMatrix FSAffineMatrixScale (
    FS\_AffineMatrix matrix,
    FS\_FLOAT xScale,
    FS\_FLOAT yScale
);
```

**Description**

Scales the matrix.

**Parameter**

matrix	[In] The input matrix.
xScale	[In] The x-direction scale coefficient.
yScale	[In] The y-direction scale coefficient.

**Return**

The result matrix.

**Head file reference**

fs\_basicTempl.h: 91

**Related method**

[FSAffineMatrixIsScale](#)

**FSAffineMatrixConcatInverse****Syntax**

```
FS_AffineMatrix FSAffineMatrixConcatInverse (
    FS\_AffineMatrix matrix,
    FS\_AffineMatrix src
);
```

**Description**

Concatenates the inverse of another matrix.

**Parameter**

matrix	[In] The input matrix.
src	[In] The input matrix, to be inversed.

**Return**

The result of a matrix concatenating another matrix inversed;

**Head file reference**

fs\_basicTempl.h: 147

**Related method**[FSAffineMatrixConcat](#)**FSAffineMatrixMatchRect****Syntax**

```
FS_AffineMatrix FSAffineMatrixMatchRect (
    FS\_FloatRect dest,
    FS\_FloatRect src
);
```

**Description**

Gets a matrix that transforms a source rectangle to dest rectangle.

**Parameter**

---

dest	[In] The dest rectangle
------	-------------------------

---

src	[In] The source rectangle.
-----	----------------------------

---

**Return**

The matrix that transforms a source rectangle to dest rectangle.

**Head file reference**

fs\_basicTempl.h: 199

**Related method**[FSAffineMatrix](#)**FSAffineMatrixGetUnitRect****Syntax**

```
FS_FloatRect FSAffineMatrixGetUnitRect (
    FS\_AffineMatrix matrix
);
```

**Description**

Gets a bounding rectangle of the parallelogram composing two unit vectors.

**Parameter**

---

matrix	[In] The input matrix.
--------	------------------------

---

**Return**

The unit rectangle.

**Head file reference**

fs\_basicTempl.h: 210

**Related method**

[FSAffineMatrixGetUnitArea](#)

### FSAffineMatrixGetUnitArea

**Syntax**

```
FS_FLOAT FSAffineMatrixGetUnitArea (
    FS\_AffineMatrix matrix
);
```

**Description**

Gets area of the parallelogram composing two unit vectors.

**Parameter**

---

matrix	[In] The input matrix.
--------	------------------------

---

**Return**

The unit area

**Head file reference**

fs\_basicTempl.h: 215

**Related method**

[FSAffineMatrixGetUnitRect](#)

### FSAffineMatrixGetXUnit

**Syntax**

```
FS_FLOAT FSAffineMatrixGetXUnit (
    FS\_AffineMatrix matrix
);
```

**Description**

Gets the x-direction unit size.

**Parameter**

---

matrix	[In] The input matrix.
--------	------------------------

---

**Return**

The x-direction unit size.

**Head file reference**

fs\_basicTempl.h: 230

**Related method**

[FSAffineMatrixGetYUnit](#)

## FSAffineMatrixGetYUnit

**Syntax**

```
FS_FLOAT FSAffineMatrixGetYUnit (
    FS\_AffineMatrix matrix
);
```

**Description**

Gets the y-direction unit size.

**Parameter**

---

matrix	[In] The input matrix.
--------	------------------------

---

**Return**

The y-direction unit size.

**Head file reference**

fs\_basicTempl.h: 235

**Related method**

[FSAffineMatrixGetXUnit](#)

## FSAffineMatrixTransformDistance

**Syntax**

```
FS_FLOAT FSAffineMatrixTransformDistance (
    FS\_AffineMatrix matrix,
    FS\_FLOAT distance
);
```

**Description**

Transforms a distance.

**Parameter**

---

matrix	[In] The input matrix.
--------	------------------------

---

---

distance	[In] The input distance.
----------	--------------------------

---

**Return**

The transformed distance.

**Head file reference**

fs\_basicTempl.h: 124

**Related method**

[FSAffineMatrixTransformRect](#)  
[FSAffineMatrixTransformPoint](#)

**FSAffineMatrixRotate****Syntax**

```
FS_AffineMatrix FSAffineMatrixRotate (
    FS\_AffineMatrix matrix,
    FS\_FLOAT fRadian,
    FS\_BOOL bPrepended
);
```

**Description**

Rotates the matrix.

**Parameter**

---

matrix	[In] The input matrix.
--------	------------------------

---

---

fRadian	[In] Rotation angle in radian.
---------	--------------------------------

---

---

bPrepended	[In] If it's TRUE, a rotation matrix is multiplied at left side, or at right side. Sets it FALSE as default.
------------	--

---

**Return**

The result matrix.

**Head file reference**

fs\_basicTempl.h: 262

**FS\_Base64Decoder**

[Return from Used by](#)

**Description**

It represents a Base64 decoder object. You can use it to decode Base64 encoded data. See [FSBase64DecoderNew](#) , [FSBase64DecoderDecode](#) .

## Returned from

[FSBase64DecoderNew](#)

## Used by

[FSBase64DecoderDecode](#)  
[FSBase64DecoderDecode2](#)  
[FSBase64DecoderDecode3](#)  
[FSBase64DecoderDestroy](#)  
[FSBase64DecoderSetDecoder](#)

## Callbacks

### Callbacks summary

[FS\\_LPBase64DecodeProc](#)

Type definition for external Base64 decoder.

### Callbacks detail

`FS_LPBase64DecodeProc`

#### Syntax

```
typedef FS_BYTEx (*FS_LPBase64DecodeProc)(  
    FS_WCHAR wch  
)
```

#### Description

Type definition for external Base64 decoder.

#### Parameter

---

wch	[In] A wide character to decode.
-----	----------------------------------

---

#### Return

A 6bits decoded byte value.

#### Head file reference

fs\_basicExpT.h: 1337

#### Related method

[FSBase64DecoderSetDecoder](#)

## Functions

## Functions summary

### [FSBase64DecoderDecode](#)

Decodes wide character array into byte buffer.

### [FSBase64DecoderDecode2](#)

Decodes wide string into a byte string.

### [FSBase64DecoderDecode3](#)

Decodes UTF-8 byte string into a byte string.

### [FSBase64DecoderDestroy](#)

Destroys the input Base64 decoder object.

### [FSBase64DecoderNew](#)

Creates the Base64 decoder object.

### [FSBase64DecoderSetDecoder](#)

Sets external Base64 decoder.

## Functions detail

### FSBase64DecoderDecode

#### Syntax

```
FS_INT32 FSBase64DecoderDecode (
    FS\_Base64Decoder decoder,
    FS\_LPCWSTR pSrc,
    FS\_INT32 iSrcLen,
    FS\_LPBYTE pDst
);
```

#### Description

Decodes wide character array into byte buffer.

#### Parameter

decoder	[In] The input Base64 decoder object.
---------	---------------------------------------

pSrc	[In] Source wide character array to decode.
------	---

iSrcLen	[In] The length of source wide character array, in wide characters.
---------	---

pDst	[Out] Destination pointer of byte data array.
------	---

#### Return

Returns the length of total data stored in destination buffer, in bytes. If *pDst* is a NULL pointer, it returns the necessary buffer size in bytes.

#### Head file reference

fs\_basicTempl.h: 2787

**Related method**[FSBase64DecoderNew](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FSBase64DecoderDecode2****Syntax**

```
FS_INT32 FSBase64DecoderDecode2 (
    FS\_Base64Decoder decoder,
    const FS\_WideString src,
    FS\_ByteString\* out_Dst
);
```

**Description**

Decodes wide string into a byte string.

**Parameter**

decoder	[In] The input Base64 decoder object.
src	[In] Source wide character string to decode.
out_Dst	[Out] Destination byte string to store decoded data.

**Return**

Returns the length of total data stored in destination, in bytes.

**Head file reference**

fs\_basicTempl.h: 2801

**Related method**[FSBase64DecoderNew](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FSBase64DecoderDecode3****Syntax**

```
FS_INT32 FSBase64DecoderDecode3 (
    FS\_Base64Decoder decoder,
    const FS\_ByteString src,
    FS\_ByteString\* out_Dst
);
```

**Description**

Decodes UTF-8 byte string into a byte string.

**Parameter**

decoder	[In] The input Base64 decoder object.
src	[In] Source UTF-8 byte string to decode.
out_Dst	[Out] Destination byte string to store decoded data.

**Return**

Returns the length of total data stored in destination, in bytes.

**Head file reference**

fs\_basicTempl.h: 2814

**Related method**

[FSBase64DecoderNew](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

## FSBase64DecoderDestroy

**Syntax**

```
void FSBase64DecoderDestroy (
    FS\_Base64Decoder decoder
);
```

**Description**

Destroys the input Base64 decoder object.

**Parameter**

decoder	[In] The input Base64 decoder object.
---------	---------------------------------------

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2759

**Related method**

[FSBase64DecoderNew](#)

**Since**  
[SDK LATEEST VERSION > 2.0](#)

## FSBase64DecoderNew

### Syntax

```
FS_Base64Decoder FSBase64DecoderNew (
    FS\_WCHAR wEqual
);
```

### Description

Creates the Base64 decoder object.

### Parameter

---

wEqual	[In] The input character used as a suffix purposes. Sets it = as default.
--------	---

---

### Return

The Base64 decoder object.

### Head file reference

[fs\\_basicTempl.h: 2753](#)

### Related method

[FSBase64DecoderDestroy](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

## FSBase64DecoderSetDecoder

### Syntax

```
void FSBase64DecoderSetDecoder (
    FS\_Base64Decoder decoder,
    FS\_LPBase64DecodeProc pDecodeProc
);
```

### Description

Sets external Base64 decoder.

### Parameter

---

decoder	[In] The input Base64 decoder object.
---------	---------------------------------------

---

---

pDecodeProc	[In] Callback function address to provide an external decoder. Sets it as NULL if need use default Base64 decoder.
-------------	---

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2775

**Related method**[FSBase64DecoderNew](#)**Since**[SDK LATEEST VERSION > 2.0](#)

## FS\_Base64Encoder

**[Return from Used by](#)****Description**

It represents a Base64 encoder object. You can use it to encode a byte data array into Base64. See [FSBase64EncoderNew](#) , [FSBase64EncoderEncode](#) .

**Returned from**[FSBase64EncoderNew](#)**Used by**

[FSBase64EncoderDestroy](#)  
[FSBase64EncoderEncode](#)  
[FSBase64EncoderEncode2](#)  
[FSBase64EncoderEncode3](#)  
[FSBase64EncoderSetEncoder](#)

**Callbacks****Callbacks summary****[FS\\_LPBase64EncodeProc](#)**

Type definition for external Base64 encoder.

**Callbacks detail****FS\_LPBase64EncodeProc****Syntax**

```
typedef FS_WCHAR (*FS_LPBase64EncodeProc)(  
    FS\_DWORD b
```

);

**Description**

Type definition for external Base64 encoder.

**Parameter**

---

b	[In] a 6bits data to encode.
---	------------------------------

---

**Return**

An encoded wide character value.

**Head file reference**

fs\_basicExpT.h: 1313

**Related method**

[FSBase64EncoderSetEncoder](#)

## Functions

### Functions summary

**[FSBase64EncoderDestroy](#)**

Destroys the input Base64 encoder object.

**[FSBase64EncoderEncode](#)**

Encodes byte data array into a wide character array.

**[FSBase64EncoderEncode2](#)**

Encodes byte string into a wide string.

**[FSBase64EncoderEncode3](#)**

Encodes byte string into a UTF-8 byte string.

**[FSBase64EncoderNew](#)**

Creates the Base64 encoder object.

**[FSBase64EncoderSetEncoder](#)**

Sets external Base64 encoder.

### Functions detail

#### [FSBase64EncoderDestroy](#)

**Syntax**

```
void FSBase64EncoderDestroy (
    FS\_Base64Encoder encoder
);
```

**Description**

Destroys the input Base64 encoder object.

**Parameter**

---

encoder	[In] The input Base64 encoder object.
---------	---------------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2677

**Related method**[FSBase64EncoderNew](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FSBase64EncoderEncode****Syntax**

```
FS_INT32 FSBase64EncoderEncode (
    FS_Base64Encoder encoder,
    FS_LPCBYTE pSrc,
    FS_INT32 iSrcLen,
    FS_LPWSTR pDst
);
```

**Description**

Encodes byte data array into a wide character array.

**Parameter**


---

encoder	[In] The input Base64 encoder object.
---------	---------------------------------------

---

pSrc	[In] Source byte data array to encode.
------	--

---

iSrcLen	[In] The length of source byte array, in bytes.
---------	---

---

pDst	[Out] Destination pointer of wide character array.
------	--

---

**Return**Returns the length of total data stored in destination buffer, in wide characters. If *pDst* is a NULL pointer, it returns the necessary buffer size in wide characters.**Head file reference**

fs\_basicTempl.h: 2705

**Related method**[FSBase64EncoderNew](#)

**Since**  
[SDK LATEEST VERSION > 2.0](#)

## FSBase64EncoderEncode2

### Syntax

```
FS_INT32 FSBase64EncoderEncode2 (
    FS\_Base64Encoder encoder,
    const FS\_ByteString src,
    FS\_WideString* out_Dst
);
```

### Description

Encodes byte string into a wide string.

### Parameter

---

encoder	[In] The input Base64 encoder object.
---------	---------------------------------------

---

src	[In] Source byte string to encode.
-----	------------------------------------

---

out_Dst	[Out] Destination wide string to store encoded data. Creates it by <a href="#">FSWideStringNew</a> .
---------	--

---

### Return

Returns the length of total data stored in destination, in wide characters.

### Head file reference

fs\_basicTempl.h: 2719

### Related method

[FSBase64EncoderNew](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

## FSBase64EncoderEncode3

### Syntax

```
FS_INT32 FSBase64EncoderEncode3 (
    FS\_Base64Encoder encoder,
    const FS\_ByteString src,
    FS\_ByteString* out_Dst
);
```

### Description

Encodes byte string into a UTF-8 byte string.

#### Parameter

---

encoder	[In] The input Base64 encoder object.
src	[In] Source byte string to encode.
out_Dst	[Out] Destination byte string to store encoded data. Creates it by <a href="#">FSByteStringNew</a> .

---

#### Return

Returns the length of total data stored in destination, in bytes.

#### Head file reference

fs\_basicTempl.h: 2732

#### Related method

[FSBase64EncoderNew](#)

#### Since

[SDK LATEEST VERSION > 2.0](#)

## FSBase64EncoderNew

#### Syntax

```
FS_Base64Encoder FSBase64EncoderNew (
    FS_WCHAR wEqual
);
```

#### Description

Creates the Base64 encoder object.

#### Parameter

---

wEqual	[In] The input character used as a suffix purposes. Sets it = as default.
--------	---

---

#### Return

The Base64 encoder object.

#### Head file reference

fs\_basicTempl.h: 2671

#### Related method

[FSBase64EncoderDestroy](#)

**Since**  
[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FSBase64EncoderSetEncoder

### Syntax

```
void FSBase64EncoderSetEncoder (
    FS\_Base64Encoder encoder,
    FS\_LPBase64EncodeProc* pEncodeProc
);
```

### Description

Sets external Base64 encoder.

### Parameter

encoder	[In] The input Base64 encoder object.
pEncodeProc	[In] Callback function address to provide an external encoder. Sets it as NULL if need use default Base64 encoder.

### Return

void

### Head file reference

fs\_basicTempl.h: 2693

### Related method

[FSBase64EncoderNew](#)

### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FS\_BinaryBuf

### [Return from Used by](#)

### Description

Dynamic binary buffers designed for more efficient appending. See [FSBinaryBufNew](#) , [FSBinaryBufDestroy](#) .

### Returned from

[FSBinaryBufNew](#)

## Used by

[FPDParserGetIndirectBinary](#)  
[FSBinaryBufAppendBlock](#)  
[FSBinaryBufAppendByte](#)  
[FSBinaryBufAppendFill](#)  
[FSBinaryBufAppendString](#)  
[FSBinaryBufAttachData](#)  
[FSBinaryBufClear](#)  
[FSBinaryBufCopyData](#)  
[FSBinaryBufDelete](#)  
[FSBinaryBufDestroy](#)  
[FSBinaryBufDetachBuffer](#)  
[FSBinaryBufEstimateSize](#)  
[FSBinaryBufGetBuffer](#)  
[FSBinaryBufGetByteString](#)  
[FSBinaryBufGetSize](#)  
[FSBinaryBufInsertBlock](#)  
[FSBinaryBufTakeOver](#)

## Functions

### Functions summary

#### [FSBinaryBufAppendBlock](#)

Appends a binary buffer block.

#### [FSBinaryBufAppendByte](#)

Appends a single byte.

#### [FSBinaryBufAppendFill](#)

Appends a byte for specified number times. Not a byte-by-byte processing, but a byte filling processing internally.

#### [FSBinaryBufAppendString](#)

Appends a non-buffered byte string.

#### [FSBinaryBufAttachData](#)

Attach to a buffer (this buffer will belong to this object). The buffer must be allocated by FS\_Alloc.

#### [FSBinaryBufClear](#)

Sets the binary buffer to be empty.

#### [FSBinaryBufCopyData](#)

Copies from another buffer.

#### [FSBinaryBufDelete](#)

Delete a inter-zone buffer defining by parameters start\_index and count in the binary buffer.

#### [FSBinaryBufDestroy](#)

Destroys the binary buffer.

#### [FSBinaryBufDetachBuffer](#)

Detaches the buffer. Just set buffer pointer to NULL, and set the binary buffer size to zero.

#### [FSBinaryBufEstimateSize](#)

Changes the allocated buffer size, and set the allocation step if *allocstep* is non-zero.

#### [FSBinaryBufGetBuffer](#)

Gets a byte pointer to the binary buffer.

#### [FSBinaryBufGetByteString](#)

Gets a byte string from the buffer.

**FSBinaryBufGetSize**

Gets the length of the binary buffer.

**FSBinaryBufInsertBlock**

Insert a binary buffer block at the specified position.

**FSBinaryBufNew**

Creates a new empty binary buffer.

**FSBinaryBufTakeOver**

Takeover another buffer.

## Functions detail

**FSBinaryBufAppendBlock****Syntax**

```
void FSBinaryBufAppendBlock (
    FS_BinaryBuf buf,
    void* pBuf,
    FS_StrSize size
);
```

**Description**

Appends a binary buffer block.

**Parameter**

---

buf	[In] The input binary buffer.
-----	-------------------------------

---

pBuf	[In] A pointer to a binary buffer block.
------	--

---

size	[In] The size in bytes of the buffer block.
------	---

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2399

**FSBinaryBufAppendByte****Syntax**

```
void FSBinaryBufAppendByte (
    FS_BinaryBuf buf,
    FS_BYT byte
);
```

**Description**

Appends a single byte.

**Parameter**

---

buf	[In] The input binary buffer.
-----	-------------------------------

---

byte	[In] A single byte.
------	---------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2431

**FSBinaryBufAppendFill****Syntax**

```
void FSBinaryBufAppendFill (
    FS\_BinaryBuf buf,
    FS\_BYTE byte,
    FS\_StrSize count
);
```

**Description**

Appends a byte for specified number times. Not a byte-by-byte processing, but a byte filling processing internally.

**Parameter**

---

buf	[In] The input binary buffer.
-----	-------------------------------

---

byte	[In] The input byte.
------	----------------------

---

count	[In] Number of times.
-------	-----------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2410

**FSBinaryBufAppendString****Syntax**

```
void FSBinaryBufAppendString (
    FS\_BinaryBuf buf,
    FS\_CHAR* str
```

);

**Description**

Appends a non-buffered byte string.

**Parameter**

buf	[In] The input binary buffer.
str	[In] A no-buffered byte string.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2421

**FSBinaryBufAttachData****Syntax**

```
void FSBinaryBufAttachData (
    FS\_BinaryBuf buf,
    void* pBuf,
    FS\_StrSize size
);
```

**Description**

Attach to a buffer (this buffer will belong to this object). The buffer must be allocated by FS\_Alloc.

**Parameter**

buf	[In] The input binary buffer.
pBuf	[In] A pointer to a binary buffer.
size	[In] The size in bytes of the buffer.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2453

## FSBinaryBufClear

### Syntax

```
void FSBinaryBufClear (
    FS\_BinaryBuf buf
);
```

### Description

Sets the binary buffer to be empty.

### Parameter

---

buf	[In] The input binary buffer.
-----	-------------------------------

---

### Return

void

### Head file reference

fs\_basicTempl.h: 2379

## FSBinaryBufCopyData

### Syntax

```
void FSBinaryBufCopyData (
    FS\_BinaryBuf buf,
    const void* pBuf,
    FS\_StrSize size
);
```

### Description

Copies from another buffer.

### Parameter

---

buf	[In] The input binary buffer.
-----	-------------------------------

---

---

pBuf	[In] The input another buffer.
------	--------------------------------

---

---

size	[In] The size in bytes of the buffer.
------	---------------------------------------

---

### Return

void

### Head file reference

fs\_basicTempl.h: 2464

## FSBinaryBufDelete

### Syntax

```
void FSBinaryBufDelete (
    FS\_BinaryBuf buf,
    FS\_INT32 startIndex,
    FS\_INT32 count
);
```

### Description

Delete a inter-zone buffer defining by parameters start\_index and count in the binary buffer.

### Parameter

buf	[In] The input binary buffer.
startIndex	[In] Specifies the zero-based index of the start position to be deleted in the binary buffer.
count	[In] Specifies the length in bytes to be deleted.

### Return

void

### Head file reference

fs\_basicTempl.h: 2486

## FSBinaryBufDestroy

### Syntax

```
void FSBinaryBufDestroy (
    FS\_BinaryBuf buf
);
```

### Description

Destroys the binary buffer.

### Parameter

buf	[In] The input binary buffer.
-----	-------------------------------

### Return

void

### Head file reference

fs\_basicTempl.h: 2370



## FSBinaryBufDetachBuffer

### Syntax

```
void FSBinaryBufDetachBuffer (
    FS\_BinaryBuf buf
);
```

### Description

Detachs the buffer. Just set buffer pointer to NULL, and set the binary buffer size to zero.

### Parameter

---

buf	[In] The input binary buffer.
-----	-------------------------------

---

### Return

void

### Head file reference

fs\_basicTempl.h: 2525

## FSBinaryBufEstimateSize

### Syntax

```
void FSBinaryBufEstimateSize (
    FS\_BinaryBuf buf,
    FS\_StrSize size,
    FS\_StrSize allocStep
);
```

### Description

Changes the allocated buffer size, and set the allocation step if *allocstep* is non-zero.

### Parameter

---

buf	[In] The input binary buffer.
-----	-------------------------------

---

---

size	[In] The new size expected.
------	-----------------------------

---

---

allocStep	[In] The new allocation step. If <i>allocstep</i> is 0, then the allocation step will not change.
-----------	---

---

### Return

void

### Head file reference

fs\_basicTempl.h: 2388

## FSBinaryBufGetBuffer

### Syntax

```
FS_LPBYTE FSBinaryBufGetBuffer (
    FS\_BinaryBuf buf
);
```

### Description

Gets a byte pointer to the binary buffer.

### Parameter

---

buf	[In] The input binary buffer.
-----	-------------------------------

---

### Return

A byte pointer to the binary buffer.

### Head file reference

fs\_basicTempl.h: 2497

## FSBinaryBufGetByteString

### Syntax

```
void FSBinaryBufGetByteString (
    FS\_BinaryBuf buf,
    FS\_ByteString* outString
);
```

### Description

Gets a byte string from the buffer.

### Parameter

---

buf	[In] The input binary buffer.
-----	-------------------------------

---

---

outString	[Out] A byte string result.
-----------	-----------------------------

---

### Return

void

### Head file reference

fs\_basicTempl.h: 2515

## FSBinaryBufGetSize

### Syntax

```
FS_StrSize FSBinaryBufGetSize (
    FS\_BinaryBuf buf
);
```

### Description

Gets the length of the binary buffer.

### Parameter

---

buf	[In] The input binary buffer.
-----	-------------------------------

---

### Return

The length in bytes of the binary buffer.

### Head file reference

fs\_basicTempl.h: 2506

## FSBinaryBufInsertBlock

### Syntax

```
void FSBinaryBufInsertBlock (
    FS\_BinaryBuf buf,
    FS\_StrSize pos,
    const void* pBuf,
    FS\_StrSize size
);
```

### Description

Insert a binary buffer block at the specified position.

### Parameter

---

buf	[In] The input binary buffer.
-----	-------------------------------

---

pos	[In] Specifies the zero-based index of the position in the binary buffer.
-----	---

---

pBuf	[In] A pointer to a binary buffer block.
------	--

---

size	[In] The size in bytes of the buffer block.
------	---

---

### Return

void

**Head file reference**

fs\_basicTempl.h: 2441

**FSBinaryBufNew****Syntax**

```
FS_BinaryBuf FSBinaryBufNew (void );
```

**Description**

Creates a new empty binary buffer.

**Return**

A new empty binary buffer.

**Head file reference**

fs\_basicTempl.h: 2361

**FSBinaryBufTakeOver****Syntax**

```
void FSBinaryBufTakeOver (
    FS\_BinaryBuf buf,
    FS\_BinaryBuf other
);
```

**Description**

Takeover another buffer.

**Parameter**

---

buf	[In] The input binary buffer.
-----	-------------------------------

---

other	[In] Another buffer
-------	---------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2475

**Note:**It attaches to the source FS\_BinaryBuf object's buffer; The source FS\_BinaryBuf object has detached the buffer.

# FS\_ByteArray

## [Return from Used by](#)

### Description

A byte array. See [FSByteArrayNew](#) , [FSByteArrayDestroy](#) .

### Returned from

#### [FSByteArrayNew](#)

### Used by

[FSByteArrayAdd](#)  
[FSByteArrayAppend](#)  
[FSByteArrayCopy](#)  
[FSByteArrayDestroy](#)  
[FSByteArrayFind](#)  
[FSByteArrayGetAt](#)  
[FSByteArrayGetDataPtr](#)  
[FSByteArrayGetSize](#)  
[FSByteArrayGetUpperBound](#)  
[FSByteArrayInsertAt](#)  
[FSByteArrayInsertNewArrayAt](#)  
[FSByteArrayRemoveAll](#)  
[FSByteArrayRemoveAt](#)  
[FSByteArraySetAt](#)  
[FSByteArraySetAtGrow](#)  
[FSByteArraySetSize](#)

### Functions

#### Functions summary

##### [FSByteArrayAdd](#)

Adds an element at the tail. Potentially grow the array

##### [FSByteArrayAppend](#)

Appends an array.

##### [FSByteArrayCopy](#)

Copies from an array.

##### [FSByteArrayDestroy](#)

Destroys a byte array.

##### [FSByteArrayFind](#)

Finds an element from specified position to last

##### [FSByteArrayGetAt](#)

Retrieves an element specified by an index number.

##### [FSByteArrayGetDataPtr](#)

Gets a pointer to the specified element in the array. Direct pointer access.

##### [FSByteArrayGetSize](#)

Gets the number of elements in the array.

##### [FSByteArrayGetUpperBound](#)

Gets the upper bound in the array, actually the maximum valid index.

#### [FSByteArrayInsertAt](#)

Inserts one or more continuous element at specified position.

#### [FSByteArrayInsertNewArrayAt](#)

Inserts an array at specified position.

#### [FSByteArrayNew](#)

Creates a new empty byte array.

#### [FSByteArrayRemoveAll](#)

Cleans up the array.

#### [FSByteArrayRemoveAt](#)

Removes a number of elements at specified position.

#### [FSByteArraySetAt](#)

Overwrites an element specified by an index number.

#### [FSByteArraySetAtGrow](#)

Sets an element value at specified position. Potentially grow the array.

#### [FSByteArraySetSize](#)

Changes the allocated size and the growing amount.

## Functions detail

### FSByteArrayAdd

#### **Syntax**

```
FS_INT32 FSByteArrayAdd (
    FS_ByteArray arr,
    FS_BYTE newItem
);
```

#### **Description**

Adds an element at the tail. Potentially grow the array

#### **Parameter**

arr	[In] The input byte array.
-----	----------------------------

newItem	[In] The input element.
---------	-------------------------

#### **Return**

The added element's index in the array.

#### **Head file reference**

fs\_basicTempl.h: 1381

### FSByteArrayAppend

#### **Syntax**

```
FS_INT32 FSByteArrayAppend (
    FS_ByteArray arr,
    const FS_ByteArray srcArr
```

);

**Description**

Appends an array.

**Parameter**

---

arr	[In] The input byte array.
-----	----------------------------

---

srcArr	[In] The input array.
--------	-----------------------

---

**Return**

The old size in elements.

**Head file reference**

fs\_basicTempl.h: 1391

## FSByteArrayCopy

**Syntax**

```
void FSByteArrayCopy (
    FS\_ByteArray arr,
    FS\_ByteArray srcArr
);
```

**Description**

Copies from an array.

**Parameter**

---

arr	[In] The input byte array.
-----	----------------------------

---

srcArr	[In] The input array.
--------	-----------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1401

## FSByteArrayDestroy

**Syntax**

```
void FSByteArrayDestroy (
    FS\_ByteArray arr
```

);

**Description**

Destroys a byte array.

**Parameter**

---

arr	[In] The input byte array.
-----	----------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1302

**FSByteArrayFind****Syntax**

```
FS_INT32 FSByteArrayFind (
    FS\_ByteArray arr,
    FS\_BYTE item,
    FS\_INT32 nstartIndex
);
```

**Description**

Finds an element from specified position to last

**Parameter**

---

arr	[In] The input byte array.
-----	----------------------------

---

---

item	[In] The input element.
------	-------------------------

---

---

nstartIndex	[In] Specifies the zero-based index of start element to find.
-------------	---

---

**Return**

An index of the found element. It can be -1 for finding none.

**Head file reference**

fs\_basicTempl.h: 1455

**FSByteArrayGetAt****Syntax**

```
FS_BYTE FSByteArrayGetAt (
```

```
FS_ByteArray arr,  
FS_INT32 index  
);
```

**Description**

Retrieves an element specified by an index number.

**Parameter**

---

arr	[In] The input byte array.
index	[In] Specifies the zero-based index of the element.

---

**Return**

An element

**Head file reference**

fs\_basicTempl.h: 1349

**FSByteArrayGetDataPtr****Syntax**

```
FS_BYTE* FSByteArrayGetDataPtr (  
    FS_ByteArray arr,  
    FS_INT32 index  
) ;
```

**Description**

Gets a pointer to the specified element in the array. Direct pointer access.

**Parameter**

---

arr	[In] The input byte array.
index	[In] Specifies the zero-based index of element in the array.

---

**Return**

A pointer to the specified element.

**Head file reference**

fs\_basicTempl.h: 1411

**FSByteArrayGetSize****Syntax**

```
FS_INT32 FSByteArrayGetSize (
    FS\_ByteArray arr
);
```

**Description**

Gets the number of elements in the array.

**Parameter**

---

arr	[In] The input byte array.
-----	----------------------------

---

**Return**

The number of elements in the array.

**Head file reference**

fs\_basicTempl.h: 1311

**FSByteArrayGetUpperBound****Syntax**

```
FS_INT32 FSByteArrayGetUpperBound (
    FS\_ByteArray arr
);
```

**Description**

Gets the upper bound in the array, actually the maximum valid index.

**Parameter**

---

arr	[In] The input byte array.
-----	----------------------------

---

**Return**

The upper bound.

**Head file reference**

fs\_basicTempl.h: 1320

**FSByteArrayInsertAt****Syntax**

```
void FSByteArrayInsertAt (
    FS\_ByteArray arr,
    FS\_INT32 index,
    FS\_BYTE newItem,
    FS\_INT32 nCount
);
```

**Description**

Inserts one or more continuous element at specified position.

**Parameter**

arr	[In] The input byte array.
index	[In] Specifies the zero-based index in the array.
newItem	[In] Specifies the element value to insert.
nCount	[In] Specifies the count of the element to insert.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1421

**FSByteArrayInsertNewArrayAt****Syntax**

```
void FSByteArrayInsertNewArrayAt (
    FS ByteArray arr,
    FS INT32 nstartIndex,
    FS ByteArray newArray
);
```

**Description**

Inserts an array at specified position.

**Parameter**

arr	[In] The input byte array.
nstartIndex	[In] Specifies the zero-based index of start element to insert at.
newArray	[In] The input array.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1444

## FSByteArrayNew

### Syntax

```
FS_ByteArray FSByteArrayNew (void );
```

### Description

Creates a new empty byte array.

### Return

A new empty byte array.

### Head file reference

fs\_basicTempl.h: 1293

## FSByteArrayRemoveAll

### Syntax

```
void FSByteArrayRemoveAll (
```

[FS\\_ByteArray](#) arr

```
 );
```

### Description

Cleans up the array.

### Parameter

---

arr	[In] The input byte array.
-----	----------------------------

---

### Return

void

### Head file reference

fs\_basicTempl.h: 1340

## FSByteArrayRemoveAt

### Syntax

```
void FSByteArrayRemoveAt (
```

[FS\\_ByteArray](#) arr,

[FS\\_INT32](#) index,

[FS\\_INT32](#) nCount

```
 );
```

### Description

Removes a number of elements at specified position.

**Parameter**

---

arr	[In] The input byte array.
-----	----------------------------

---

index	[In] Specifies the zero-based index in the array.
-------	---

---

nCount	[In] Specifies the count of element to remove.
--------	--

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1433

**FSByteArraySetAt****Syntax**

```
void FSByteArraySetAt (
    FS\_ByteArray arr,
    FS\_INT32 index,
    FS\_BYTE newItem
);
```

**Description**

Overwrites an element specified by an index number.

**Parameter**

---

arr	[In] The input byte array.
-----	----------------------------

---

index	[In] Specifies the zero-based index of the element.
-------	---

---

newItem	[In] An element
---------	-----------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1359

**FSByteArraySetAtGrow**

**Syntax**

```
void FSByteArraySetAtGrow (
    FS ByteArray arr,
    FS INT32 index,
    FS BYTE newItem
);
```

**Description**

Sets an element value at specified position. Potentially grow the array.

**Parameter**

---

arr	[In] The input byte array.
-----	----------------------------

---

index	[In] Specifies the zero-based index of element in the array.
-------	--

---

newItem	[In] The input element.
---------	-------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1370

**FSByteArraySetSize****Syntax**

```
void FSByteArraySetSize (
    FS ByteArray arr,
    FS INT32 nNewSize,
    FS INT32 nGrowBy
);
```

**Description**

Changes the allocated size and the growing amount.

**Parameter**

---

arr	[In] The input byte array.
-----	----------------------------

---

nNewSize	[In] The new size in elements expected.
----------	---

---

nGrowBy	[In] The growing amount in elements expected. <i>nGrowBy</i> can be -1 for the growing amount unchanged.
---------	--

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1329

## FS\_ByteString

**[Return from Used by](#)**

### Description

An object preparing a byte buffer. You can use this object to store a byte stream, but you can use it with a character string. See [FSByteStringNew](#) , [FSByteStringNew2](#) , [FSByteStringNew3](#) , [FSByteStringNew4](#) , [FSByteStringDestroy](#) .

### Returned from

[FSByteStringArrayGetDataPtr](#)  
[FSByteStringArrayNew](#)  
[FSByteStringNew](#)  
[FSByteStringNew2](#)  
[FSByteStringNew3](#)  
[FSByteStringNew4](#)

### Used by

[FSBase64DecoderDecode2](#)  
[FSBase64DecoderDecode3](#)  
[FSBase64EncoderEncode3](#)  
[FSBinaryBufGetByteString](#)  
[FSByteStringArrayAdd](#)  
[FSByteStringArrayAdd2](#)  
[FSByteStringArrayDestroy](#)  
[FSByteStringArrayGetAt](#)  
[FSByteStringArrayGetDataPtr](#)  
[FSByteStringArrayGetSize](#)  
[FSByteStringArrayRemoveAll](#)  
[FSByteStringArrayRemoveAt](#)  
[FSCodeTransformationDecodeText](#)  
[FSCodeTransformationEncodeString](#)  
[FSCodeTransformationEncodeText](#)  
[FSCodeTransformationNameDecode](#)  
[FSCodeTransformationNameEncode](#)  
[FSGUIDToString](#)  
[FSMapByteStringToPtrGetNextAssoc](#)  
[FSMapByteStringToPtrHashKey](#)  
[FSMapByteStringToPtrLookup](#)  
[FSMapByteStringToPtrRemoveKey](#)  
[FSMapByteStringToPtrSetAt](#)  
[FSUTF8EncoderAppendStr](#)  
[FSUUIDGenerate](#)  
[FSUUIDGenerateTime](#)

[FSUUIDGenerateRandom](#)  
[FSUUIDSetTsPath](#)  
[FSUUIDSetState](#)  
[FSUUIDSetUserData](#)  
[FSWideStringConvertFrom](#)  
[FSWideStringFromLocal2](#)  
[FSWideStringUTF16LE\\_Encode](#)  
[FSWideStringUTF8Encode](#)  
[FSXMLElementGetAttrByIndex](#)  
[FSXMLElementGetNamespace](#)  
[FSXMLElementGetNamespaceURI](#)  
[FSXMLElementGetTagName](#)  
[FSXMLElementOutputStream](#)  
[FDRMCategoriesReadCountSubCategories](#)  
[FDRMCategoriesReadDestroy](#)  
[FRAnnotGetSubType](#)  
[FRAnnotGetType](#)  
[FRAppGetEditionType](#)  
[FRAppGetName](#)  
[FRAppGetNavPanelNameByIndex](#)  
[FRAppGetOEMVersion](#)  
[FRFuncBtnGetName](#)  
[FRMenuItemGetName](#)  
[FRRibbonCategoryGetName](#)  
[FRRibbonElementGetCategoryName](#)  
[FRRibbonElementGetName](#)  
[FRRibbonElementGetPanelName](#)  
[FRRibbonListButtonGetSelectedItem](#)  
[FRRibbonPaletteButtonGetItemInfo](#)  
[FRRibbonPaletteButtonGetSelectedItem](#)  
[FRRibbonPaletteButtonSetItemAccNameTitle](#)  
[FRRibbonPanelGetName](#)  
[FRToolGetName](#)  
[FRToolBarGetName](#)  
[FRToolButtonGetName](#)  
[FRWindowsDIBGetBitmapInfo](#)  
[FPDFDFDocWriteBuf](#)  
[FPDACTIONGetNameAction](#)  
[FPDACTIONGetTypeAction](#)  
[FPDACTIONGetURI](#)  
[FPDAnnotGetSubType](#)  
[FPDArrayGetString](#)  
[FPDCConnectedInfoGetEncryptEnvelope](#)  
[FPDCConnectedInfoGetEndpoint](#)  
[FPDCConnectedInfoGetId](#)  
[FPDCConnectedInfoSetEncryptEnvelope](#)  
[FPDCConnectedInfoSetId](#)  
[FPDDEFAULTAPPEARANCEGetColorString](#)  
[FPDDEFAULTAPPEARANCEGetFont](#)  
[FPDDEFAULTAPPEARANCEGetFontString](#)  
[FPDDEFAULTAPPEARANCEGetTextMatrixString](#)  
[FPDDestGetRemoteName](#)  
[FPDDictionaryGetNextElement](#)  
[FPDDictionaryGetString](#)  
[FPDDictionarySetAtString](#)  
[FPDDocGetID](#)  
[FPDFFontAppendChar2](#)

[FPDFFontEncodeString](#)  
[FPDFFontFXFontGetFaceName](#)  
[FPDFFontFXFontGetFamilyName](#)  
[FPDFFontGetBaseFont](#)  
[FPDFFontGetFontTypeNames](#)  
[FPDFFormFindCSName](#)  
[FPDFFormFindFontName](#)  
[FPDFFormRealizeResource](#)  
[FPDFFormControlGetCheckedAPState](#)  
[FPDFFormFieldGetDefaultStyle](#)  
[FPDInterFormAddFormFont](#)  
[FPDInterFormAddNativeFormFont](#)  
[FPDInterFormAddNativeFormFont2](#)  
[FPDInterFormFindFormFont](#)  
[FPDInterFormFindFormFont2](#)  
[FPDInterFormFindFormFont3](#)  
[FPDInterFormGenerateNewResourceName](#)  
[FPDInterFormGetFormFont](#)  
[FPDInterFormGetFormFont3](#)  
[FPDInterFormGetNativeFont](#)  
[FPDInterFormGetNativeFont2](#)  
[FPDInterFormGetNativeFormFont](#)  
[FPDInterFormGetNativeFormFont2](#)  
[FPDLWinParamGetDefaultDirectory](#)  
[FPDLWinParamGetFileName](#)  
[FPDLWinParamGetOperation](#)  
[FPDLWinParamGetParam](#)  
[FPDMediaPlayerGetOSArray](#)  
[FPDMediaPlayerGetSoftwareURI](#)  
[FPDNameGetString](#)  
[FPDNumberGetString](#)  
[FPDOBJArchiveLoaderLoadByteString](#)  
[FPDOBJArchiveSaverSaveByteString](#)  
[FPDOBJECTGetString](#)  
[FPDOCGroupGetCreatorInfo](#)  
[FPDOCGroupGetLanguageInfo](#)  
[FPDOCGroupGetPageElementType](#)  
[FPDOCGroupGetPrintInfo](#)  
[FPDOCGroup GetUser Type](#)  
[FPDPAGEFindCSName](#)  
[FPDPAGEFindFontName](#)  
[FPDPAGEGetPageText](#)  
[FPDPAGERealizeResource](#)  
[FPDPARSERGetPassword](#)  
[FPDPARSERGetStandardSecurityUserPassword](#)  
[FPDRenderDeviceDrawTextString2](#)  
[FPDRenditionGetMediaBaseURL](#)  
[FPDRenditionGetMediaClipContentType](#)  
[FPDStringGetString](#)  
[FPDStringNew](#)  
[FPDTextObjectSetText2](#)  
[FSByteStringCastToLPCBYTE](#)  
[FSByteStringCastToLPCSTR](#)  
[FSByteStringCompare](#)  
[FSByteStringConcat](#)  
[FSByteStringConcat2](#)  
[FSByteStringConvertFrom](#)

[FSByteStringCopy](#)  
[FSByteStringDelete](#)  
[FSByteStringDestroy](#)  
[FSByteStringEmpty](#)  
[FSByteStringEqual](#)  
[FSByteStringEqualNoCase](#)  
[FSByteStringFill](#)  
[FSByteStringFind](#)  
[FSByteStringFind2](#)  
[FSByteStringFormat](#)  
[FSByteStringFormatFloat](#)  
[FSByteStringFormatInteger](#)  
[FSByteStringFormatV](#)  
[FSByteStringFromUnicode](#)  
[FSByteStringFromUnicode2](#)  
[FSByteStringGetAt](#)  
[FSByteStringGetID](#)  
[FSByteStringGetLength](#)  
[FSByteStringInsert](#)  
[FSByteStringIsEmpty](#)  
[FSByteStringLeft](#)  
[FSByteStringMakeLower](#)  
[FSByteStringMakeUpper](#)  
[FSByteStringMid](#)  
[FSByteStringMid2](#)  
[FSByteStringRemove](#)  
[FSByteStringReplace](#)  
[FSByteStringReserve](#)  
[FSByteStringRight](#)  
[FSByteStringSetAt](#)  
[FSByteStringTrimLeft](#)  
[FSByteStringTrimLeft2](#)  
[FSByteStringTrimLeft3](#)  
[FSByteStringTrimRight](#)  
[FSByteStringTrimRight2](#)  
[FSByteStringTrimRight3](#)  
[FSByteStringUTF8Decode](#)

## Functions

### Functions summary

#### [FSByteStringCastToLPCBYTE](#)

Casts the byte string to byte buffer.

#### [FSByteStringCastToLPCSTR](#)

cast the byte string to char buffer.

#### [FSByteStringCompare](#)

Compares the the string with another string, case-sensitive.

#### [FSByteStringConcat](#)

Concatenates a source byte string.

#### [FSByteStringConcat2](#)

Concatenates a normal string to byte string.

#### [FSByteStringConvertFrom](#)

Loads unicode data into this byte string, using specified character mapper. If no character mapper specified, the system default mapper will be used.

**FSByteStringCopy**

Copies from a source byte string.

**FSByteStringDelete**

Deletes one or more characters starting from specific position.

**FSByteStringDestroy**

Destroys the byte string.

**FSByteStringEmpty**

Sets this string to be empty.

**FSByteStringEqual**

Checks if this string equals to another.

**FSByteStringEqualNoCase**

Checks if two string equal not considering case.

**FSByteStringFill**

Fills a normal string to byte string.

**FSByteStringFind**

Finds a sub-string, from specific position. Only first occurrence is found.

**FSByteStringFind2**

Finds a character, from specific position. Only first occurrence is found.

**FSByteStringFormat**

Formats a number of parameters into this byte string.

**FSByteStringFormatFloat**

Converts from floating-point number.

**FSByteStringFormatInteger**

Converts from Integer.

**FSByteStringFormatV**

Formats a number of parameters into this byte string. using va\_list.

**FSByteStringFromUnicode**

Converts from Unicode to system multi-byte charset.

**FSByteStringFromUnicode2**

Converts from Unicode to system multi-byte charset.

**FSByteStringGetAt**

Retrieves a single byte specified by an index number.

**FSByteStringGetID**

Gets a DWORD identifier of the string, from a particular position.

**FSByteStringGetLength**

Gets number of bytes in the byte string (not counting any possible terminator).

**FSByteStringInsert**

Inserts a character before specific position.

**FSByteStringIsEmpty**

Determines whether it is empty or not.

**FSByteStringLeft**

Extracts the leftmost count bytes as a substring.

**FSByteStringMakeLower**

Changes case of English letters to lower.

**FSByteStringMakeUpper**

Changes case of English letters to upper.

**FSByteStringMid**

Extracts a substring starting at position nFirst (zero-based) to last.

**FSByteStringMid2**

Extracts a substring starting at position nFirst (zero-based) to position *first + count*.

**FSByteStringNew**

Creates a new empty byte string.

**FSByteStringNew2**

Creates a new byte string from a single character.

**FSByteStringNew3**

Creates a new byte string from a character string.

**FSByteStringNew4**

Creates a new byte string from a byte string.

**FSByteStringRemove**

Removes all occurrence of a particular character.

**FSByteStringReplace**

Replace all patterns in the string with a new sub-string.

**FSByteStringReserve**

Reserves a buffer that can hold specific number of bytes.

**FSByteStringRight**

Extracts the rightmost count as a substring.

**FSByteStringSetAt**

Overwrites a single byte specified by an index number.

**FSByteStringTrimLeft**

Trims white spaces from the left side of the byte string.

**FSByteStringTrimLeft2**

Trims continuous occurrences of specified characters from left side of the byte string.

**FSByteStringTrimLeft3**

Trims continuous occurrences of specified characters from left side of the byte string.

**FSByteStringTrimRight**

Trims white spaces from the right side of the byte string.

**FSByteStringTrimRight2**

Trims continuous occurrences of specified character from right side of the byte string.

**FSByteStringTrimRight3**

Trims continuous occurrences of specified characters from right side of the byte string.

**FSByteStringUTF8Decode**

Decode a UTF-8 unicode string (assume this byte string is UTF-8 encoded)

## Functions detail

### FSByteStringCastToLPCBYTE

**Syntax**

```
FS_LPCBYTE FSByteStringCastToLPCBYTE (
    FS_ByteString str
);
```

**Description**

Casts the byte string to byte buffer.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

**Return**

The casted byte buffer.

**Head file reference**

fs\_stringTempl.h: 547

**Since**

[SDK LATEEST VERSION > 1.0](#)

## FSByteStringCastToLPCSTR

**Syntax**

```
FS_LPCSTR FSByteStringCastToLPCSTR (
    FS\_ByteString str
);
```

**Description**

cast the byte string to char buffer.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

**Return**

The casted char buffer.

**Head file reference**

fs\_stringTempl.h: 517

## FSByteStringCompare

**Syntax**

```
FS_INT32 FSByteStringCompare (
    FS\_ByteString str1,
    const FS\_ByteString str2
);
```

**Description**

Compares the the string with another string, case-sensitive.

**Parameter**

---

str1	[In] The input byte string.
------	-----------------------------

---

str2	[In] The byte string to be compared.
------	--------------------------------------

---

**Return**

-1 if this string is "smaller" (in alphabetic order) than the other, 0 for equal, 1 for larger in alphabetic order.

**Head file reference**

fs\_stringTempl.h: 161

**FSByteStringConcat****Syntax**

```
void FSByteStringConcat (
    FS\_ByteString str,
    const FS\_ByteString src
);
```

**Description**

Concatenates a source byte string.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

src	[In] The source byte string.
-----	------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 213

**FSByteStringConcat2****Syntax**

```
void FSByteStringConcat2 (
    FS\_ByteString str,
    FS\_LPCSTR src
);
```

**Description**

Concatenates a normal string to byte string.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

src	[In] The source normal string.
-----	--------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 223

**FSByteStringConvertFrom****Syntax**

```
void FSByteStringConvertFrom (
    FS\_ByteString str,
    FS\_WideString wstr,
    FS\_CharMap pCharMap
);
```

**Description**

Loads unicode data into this byte string, using specified character mapper. If no character mapper specified, the system default mapper will be used.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

wstr	[In] A wide string.
------	---------------------

---

pCharMap	[In] A character mapper. Invokes <a href="#">FSCharMapGetDefaultMapper</a> to get the default mapper. Sets it NULL as default.
----------	---

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 557

**Since**

[SDK LATEEST VERSION > 7.3.1](#)

**FSByteStringCopy****Syntax**

```
void FSByteStringCopy (
    FS\_ByteString str,
    const FS\_ByteString src
);
```



**Description**

Copies from a source byte string.

**Parameter**

---

str	[In] The input byte string.
src	[In] The source byte string.

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 193

**FSByteStringDelete****Syntax**

```
FS_StrSize FSByteStringDelete (
    FS_ByteString str,
    FS_StrSize nIndex,
    FS_StrSize count
);
```

**Description**

Deletes one or more characters starting from specific position.

**Parameter**

---

str	[In] The input byte string.
nIndex	[In] Specifies the zero-based index in the byte string for starting deleting.
count	[In] Count of bytes to be deleted.

---

**Return**

The new length of the byte string.

**Head file reference**

fs\_stringTempl.h: 274

**FSByteStringDestroy****Syntax**

```
void FSByteStringDestroy (
    FS\_ByteString str
);
```

**Description**

Destroys the byte string.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

**Return**

void

**Head file reference**

[fs\\_stringTempl.h](#): 134

**FSByteStringEmpty****Syntax**

```
void FSByteStringEmpty (
    FS\_ByteString str
);
```

**Description**

Sets this string to be empty.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

**Return**

void

**Head file reference**

[fs\\_stringTempl.h](#): 233

**FSByteStringEqual****Syntax**

```
FS_BOOL FSByteStringEqual (
    FS\_ByteString str1,
    const FS\_ByteString str2
);
```

**Description**

Checks if this string equals to another.

#### Parameter

str1	[In] The input byte string.
str2	[In] The byte string to be compared.

#### Return

TRUE means equal, otherwise not equal.

#### Head file reference

fs\_stringTempl.h: 172

**Note:** It's faster than Compares if you just want to check whether two strings equal.

## FSByteStringEqualNoCase

#### Syntax

```
FS_BOOL FSByteStringEqualNoCase (
    FS_ByteString str1,
    const FS_ByteString str2
);
```

#### Description

Checks if two string equal not considering case. It means letters 'A'-'Z' will be considered same as 'a'-'z'.

#### Parameter

str1	[In] The input byte string.
str2	[In] Byte string to be compared.

#### Return

TRUE means equal, otherwise not equal.

#### Head file reference

fs\_stringTempl.h: 182

## FSByteStringFill

#### Syntax

```
void FSByteStringFill (
    FS_ByteString str,
    FS_LPCSTR src
```

);

**Description**

Fills a normal string to byte string.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

src	[In] The source normal string.
-----	--------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 203

**FSByteStringFind****Syntax**

```
FS_StrSize FSByteStringFind (
    FS_ByteString str,
    const FS_ByteString strSub,
    FS_StrSize start
);
```

**Description**

Finds a sub-string, from specific position. Only first occurrence is found.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

strSub	[In] The sub-string to be found.
--------	----------------------------------

---

start	[In] Specifies the zero-based index of the starting position to do finding.
-------	---

---

**Return**

-1:Not found.

other value: Specifies position in the string.

**Head file reference**

fs\_stringTempl.h: 360

**FSByteStringFind2****Syntax**

```
FS_StrSize FSByteStringFind2 (
    FS\_ByteString str,
    FS\_CHAR ch,
    FS\_StrSize start
);
```

**Description**

Finds a character, from specific position. Only first occurrence is found.

**Parameter**

str	[In] The input byte string.
ch	[In] The character to be found.
start	[In] Specifies the zero-based index of the starting position to do finding.

**Return**

-1: Not found. other value: Specifies position in the string.

**Head file reference**

fs\_stringTempl.h: 372

**FSByteStringFormat****Syntax**

```
void FSByteStringFormat (
    FS\_ByteString str,
    FS\_LPCSTR lpszFormat,
    ...
);
```

**Description**

Formats a number of parameters into this byte string.

**Parameter**

str	[In] The input byte string.
lpszFormat	[In] Specifies a format-control string.

---

... [In] format arguments list.

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 285

**Note:** On desktop platforms, this function supports all the sprintf() formats. On embedded platforms, it supports only a subset of formats:

- Supported types: d, u, f, g, x, X, s, c, %.
- 
- Width field supported;
- 
- Precision not supported;
- 
- Flags supported: '0';

**FSByteStringFormatFloat****Syntax**

```
void FSByteStringFormatFloat (
    FS_FLOAT f,
    FS_ByteString* outResult
);
```

**Description**

Converts from floating-point number.

**Parameter**

---

f	[In] The input floating-point number.
---	---------------------------------------

---

---

outResult	[Out] A result byte string.
-----------	-----------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 507

## FSByteStringFormatInteger

### Syntax

```
void FSByteStringFormatInteger (
    FS\_INT32 i,
    FS\_DWORD flags,
    FS\_ByteString* outResult
);
```

### Description

Converts from Integer.

### Parameter

---

i	[In] The input integer
---	------------------------

---

flags	[In] The formating flags.
-------	---------------------------

---

outResult	[Out] A result byte string.
-----------	-----------------------------

---

### Return

void

### Head file reference

fs\_stringTempl.h: 493

**Note:** The flags can be following flags (single or compound): - FS\_FORMAT\_SIGNED - FS\_FORMAT\_HEX - FS\_FORMAT\_CAPITAL

## FSByteStringFormatV

### Syntax

```
void FSByteStringFormatV (
    FS\_ByteString str,
    FS\_LPCSTR lpszFormat,
    va\_list argList
);
```

### Description

Formats a number of parameters into this byte string, using va\_list.

### Parameter

---

str	[In] The input byte string.
-----	-----------------------------

---

---

<code>lpszFormat</code>	[In] Specifies a format-control string.
-------------------------	---

---

<code>argList</code>	[In] Variable-argument lists.
----------------------	-------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 536

**FSByteStringFromUnicode****Syntax**

```
void FSByteStringFromUnicode (
    const FS_LPWSTR src,
    FS_StrSize len,
    FS_ByteString* outStr
);
```

**Description**

Converts from Unicode to system multi-byte charset.

**Parameter**


---

<code>src</code>	[In] Pointer to a Unicode string.
------------------	-----------------------------------

---

<code>len</code>	[In] The length of the Unicode string. len can be -1 for zero terminated Unicode string.
------------------	--

---

<code>outStr</code>	[Out] The result byte string.
---------------------	-------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 113

**FSByteStringFromUnicode2****Syntax**

```
void FSByteStringFromUnicode2 (
    const FS_WideString src,
    FS_ByteString* outStr
);
```

**Description**

Converts from Unicode to system multi-byte charset.

**Parameter**

---

src	[In] Pointer to a Unicode string.
-----	-----------------------------------

---

outStr	[Out] The result byte string.
--------	-------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 124

**FSByteStringGetAt****Syntax**

```
FS_BYTFSByteStringGetAt (
    FS_ByteString str,
    FS_StrSize nIndex
);
```

**Description**

Retrieves a single byte specified by an index number.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

nIndex	[In] Specifies the zero-based index in the byte string.
--------	---

---

**Return**

A single byte.

**Head file reference**

fs\_stringTempl.h: 242

**FSByteStringGetID****Syntax**

```
FS_DWORD FSByteStringGetID (
    FS_ByteString str,
    FS_StrSize startPos
);
```

**Description**

Gets a DWORD identifier of the string, from a particular position. This DWORD can be used for quick comparison. Using MSB-first scheme. If the string doesn't have enough bytes, then zero will be used missing bytes.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

startPos	[In] Start position in the byte string.
----------	---

---

**Return**

A DWORD identifier of the string, from a particular position.

**Head file reference**

fs\_stringTempl.h: 481

**FSByteStringGetLength****Syntax**

```
FS_StrSize FSByteStringGetLength (
    FS\_ByteString str
);
```

**Description**

Gets number of bytes in the byte string (not counting any possible terminator).

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

**Return**

The Length of the byte string.

**Head file reference**

fs\_stringTempl.h: 143

**FSByteStringInsert****Syntax**

```
FS_StrSize FSByteStringInsert (
    FS\_ByteString str,
    FS\_StrSize nIndex,
    FS\_CHAR ch
);
```

**Description**

Inserts a character before specific position.

**Parameter**

---

str	[In] The input byte string.
nIndex	[In] Specifies the zero-based index in the byte string.
ch	[In] A single character.

---

**Return**

The new length of the byte string.

**Head file reference**

fs\_stringTempl.h: 263

**FSByteStringIsEmpty****Syntax**

```
FS_BOOL FSByteStringIsEmpty (
    FS\_ByteString str
);
```

**Description**

Determines whether it is empty or not.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

**Return**

[TRUE](#) means empty, otherwise not empty.

**Head file reference**

fs\_stringTempl.h: 152

**FSByteStringLeft****Syntax**

```
void FSByteStringLeft (
    FS\_ByteString str,
    FS\_StrSize count,
    FS\_ByteString* outStr
);
```

**Description**

Extracts the leftmost count bytes as a substring.

**Parameter**

str	[In] The input byte string.
count	[In] The count of bytes expected to extract for the substring.
outStr	[In] A leftmost substring.

**Return**

void

**Head file reference**

fs\_stringTempl.h: 338

**FSByteStringMakeLower****Syntax**

```
void FSByteStringMakeLower (
    FS\_ByteString str
);
```

**Description**

Changes case of English letters to lower.

**Parameter**

str	[In] The input byte string.
-----	-----------------------------

**Return**

void

**Head file reference**

fs\_stringTempl.h: 384

**FSByteStringMakeUpper****Syntax**

```
void FSByteStringMakeUpper (
    FS\_ByteString str
);
```

**Description**

Changes case of English letters to upper.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 393

**FSByteStringMid****Syntax**

```
void FSByteStringMid (
    FS\_ByteString str,
    FS\_StrSize first,
    FS\_ByteString* outStr
);
```

**Description**

Extracts a substring starting at position nFirst (zero-based) to last.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

first	[In] Specifies the zero-based index of the starting position in the byte string.
-------	--

---

outStr	[Out] A substring.
--------	--------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 315

**FSByteStringMid2****Syntax**

```
void FSByteStringMid2 (
    FS\_ByteString str,
```

---

```
FS_StrSize first,
FS_StrSize count,
FS_ByteString* outStr
);
```

**Description**

Extracts a substring starting at position nFirst (zero-based) to position *first + count* .

**Parameter**


---

str	[In] The input byte string.
first	[In] Specifies the zero-based index of the starting position in the byte string.
count	[In] The count of bytes expected to extract for the sub-string.
outStr	[In] A substring.

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 326

**FSByteStringNew****Syntax**

```
FS_ByteString FSByteStringNew (void );
```

**Description**

Creates a new empty byte string.

**Return**

A new empty byte string.

**Head file reference**

fs\_stringTempl.h: 75

**FSByteStringNew2****Syntax**

```
FS_ByteString FSByteStringNew2 (
    FS_CHAR ch
);
```

**Description**

Creates a new byte string from a single character.

**Parameter**

---

ch	[In] A single character.
----	--------------------------

---

**Return**

A new byte string.

**Head file reference**

fs\_stringTempl.h: 84

**FSByteStringNew3****Syntax**

```
FS_ByteString FSByteStringNew3 (
    FS_LPCSTR ptr,
    FS_StrSize len
);
```

**Description**

Creates a new byte string from a character string.

**Parameter**

---

ptr	[In] Pointer to a character string
-----	------------------------------------

---

len	[In] The length of the character string. len can be -1 for zero terminated string.
-----	--

---

**Return**

A new byte string.

**Head file reference**

fs\_stringTempl.h: 93

**FSByteStringNew4****Syntax**

```
FS_ByteString FSByteStringNew4 (
    FS_LPCBYTE ptr,
    FS_StrSize len
);
```

**Description**

Creates a new byte string from a byte string.

**Parameter**

ptr	[In] Pointer to a byte string.
len	[In] The length of the byte string.

**Return**

A new byte string.

**Head file reference**

fs\_stringTempl.h: 103

**FSByteStringRemove****Syntax**

```
FS_StrSize FSByteStringRemove (
    FS_ByteString str,
    FS_CHAR ch
);
```

**Description**

Removes all occurrence of a particular character.

**Parameter**

str	[In] The input byte string.
ch	[In] Specified the character to be removed.

**Return**

The number of characters removed.

**Head file reference**

fs\_stringTempl.h: 471

**FSByteStringReplace****Syntax**

```
FS_StrSize FSByteStringReplace (
    FS_ByteString str,
    const FS_ByteString strOld,
    const FS_ByteString strNew
```

);

**Description**

Replace all patterns in the string with a new sub-string.

**Parameter**

str	[In] The input byte string.
strOld	[In] Specified the string to be matched and replaced in the byte string.
strNew	[In] Specified the string to replace.

**Return**

The number of replaced patterns.

**Head file reference**

fs\_stringTempl.h: 460

**FSByteStringReserve****Syntax**

```
void FSByteStringReserve (
    FS\_ByteString str,
    FS\_StrSize len
);
```

**Description**

Reserves a buffer that can hold specific number of bytes.

**Parameter**

str	[In] The input byte string.
len	[In] The Length expected to reserve.

**Return**

void

**Head file reference**

fs\_stringTempl.h: 303

**Note:**The content of this string won't be changed. This can be used if application anticipates the string may grow many times, in this case, reserving a larger buffer will support string growth without buffer reallocation.

## FSByteStringRight

### Syntax

```
void FSByteStringRight (
    FS\_ByteString str,
    FS\_StrSize count,
    FS\_ByteString* outStr
);
```

### Description

Extracts the rightmost count as a substring.

### Parameter

---

str	[In] The input byte string.
-----	-----------------------------

---

count	[In] The count of bytes expected to extract for the substring.
-------	--

---

outStr	[In] A rightmost substring.
--------	-----------------------------

---

### Return

void

### Head file reference

fs\_stringTempl.h: 349

## FSByteStringSetAt

### Syntax

```
void FSByteStringSetAt (
    FS\_ByteString str,
    FS\_StrSize nIndex,
    FS\_CHAR ch
);
```

### Description

Overwrites a single byte specified by an index number.

### Parameter

---

str	[In] The input byte string.
-----	-----------------------------

---

---

nIndex	[In] Specifies the zero-based index in the byte string.
--------	---

---

ch	[In] A single character.
----	--------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 252

**FSByteStringTrimLeft****Syntax**

```
void FSByteStringTrimLeft (
    FS_ByteString str
);
```

**Description**

Trims white spaces from the left side of the byte string.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 431

**FSByteStringTrimLeft2****Syntax**

```
void FSByteStringTrimLeft2 (
    FS_ByteString str,
    FS_CHAR chTarget
);
```

**Description**

Trims continuous occurrences of specified characters from left side of the byte string.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

---

chTarget	[In] The specified character.
----------	-------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 440

**FSByteStringTrimLeft3****Syntax**

```
void FSByteStringTrimLeft3 (
    FS\_ByteString str,
    FS\_ByteString szTargets
);
```

**Description**

Trims continuous occurrences of specified characters from left side of the byte string.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

---

szTargets	[In] The specified characters.
-----------	--------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 450

**FSByteStringTrimRight****Syntax**

```
void FSByteStringTrimRight (
    FS\_ByteString str
);
```

**Description**

Trims white spaces from the right side of the byte string.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 402

**FSByteStringTrimRight2****Syntax**

```
void FSByteStringTrimRight2 (
    FS\_ByteString str,
    FS\_CHAR chTarget
);
```

**Description**

Trims continuous occurrences of specified character from right side of the byte string.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

chTarget	[In] The specified character.
----------	-------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 411

**FSByteStringTrimRight3****Syntax**

```
void FSByteStringTrimRight3 (
    FS\_ByteString str,
    FS\_ByteString szTargets
);
```

**Description**

Trims continuous occurrences of specified characters from right side of the byte string.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

szTargets	[In] The specified characters.
-----------	--------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 421

**FSByteStringUTF8Decode****Syntax**

```
void FSByteStringUTF8Decode (
    FS\_ByteString str,
    FS\_WideString* outWstr
);
```

**Description**

Decode a UTF-8 unicode string (assume this byte string is UTF-8 encoded)

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

outWstr	[Out] The result wide string.
---------	-------------------------------

---

**Return**

A unicode string.

**Head file reference**

fs\_stringTempl.h: 526

## FS\_ByteStringArray

**Return from Used by****Description**

A byte string array. See [FSByteArrayNew](#) , [FSByteArrayDestroy](#) .

**Returned from****FSByteArrayNew****Used by**

[FPDMediaPlayerGetOSArray](#)

[FPDPageGetPageText](#)

[FPDTTextObjectSetText2](#)

[FSByteArrayAdd](#)

[\*\*FSByteStringArrayAdd\*\*](#)  
[\*\*FSByteStringArrayDestroy\*\*](#)  
[\*\*FSByteStringArrayGetAt\*\*](#)  
[\*\*FSByteStringArrayGetDataPtr\*\*](#)  
[\*\*FSByteStringArrayGetSize\*\*](#)  
[\*\*FSByteStringArrayRemoveAll\*\*](#)  
[\*\*FSByteStringArrayRemoveAt\*\*](#)

## Functions

### Functions summary

#### [\*\*FSByteStringArrayAdd\*\*](#)

Adds an element at the tail. Potentially growing the array

#### [\*\*FSByteStringArrayAdd2\*\*](#)

Add a copy of an existing object to the array.

#### [\*\*FSByteStringArrayDestroy\*\*](#)

Destroys the byte-string array.

#### [\*\*FSByteStringArrayGetAt\*\*](#)

Retrieves an element specified by an index number.

#### [\*\*FSByteStringArrayGetDataPtr\*\*](#)

The pointer to the specified element.

#### [\*\*FSByteStringArrayGetSize\*\*](#)

Gets the number of elements in the array.

#### [\*\*FSByteStringArrayNew\*\*](#)

Creates a new empty byte-string array.

#### [\*\*FSByteStringArrayRemoveAll\*\*](#)

Cleans up the array.

#### [\*\*FSByteStringArrayRemoveAt\*\*](#)

Removes a number of elements at specified position.

### Functions detail

#### FSByteStringArrayAdd

##### Syntax

```
void FSByteStringArrayAdd (
    FS\_ByteStringArray arr,
    FS\_LPSTR newItem,
    newItem
);
```

##### Description

Adds an element at the tail. Potentially growing the array

##### Parameter

---

arr	[In] The input byte-string array.
-----	-----------------------------------

---

newItem	[In] The input element.
---------	-------------------------

---

---

newItem	[In] The length.
---------	------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1883

**FSByteStringArrayAdd2****Syntax**

```
void FSByteStringArrayAdd2 (
    FS\_ByteStringArray arr,
    FS\_ByteString bsNew
);
```

**Description**

Add a copy of an existing object to the array.

**Parameter**

---

arr	[In] The input byte-string array.
-----	-----------------------------------

---

bsNew	[In] The new byte string.
-------	---------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1904

**FSByteStringArrayDestroy****Syntax**

```
void FSByteStringArrayDestroy (
    FS\_ByteStringArray arr
);
```

**Description**

Destroys the byte-string array.

**Parameter**

---

arr	[In] The input byte-string array.
-----	-----------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1845

**FSByteStringArrayGetAt****Syntax**

```
void FSByteStringArrayGetAt (
    FS\_ByteStringArray arr,
    FS\_INT32 index,
    FS\_ByteString* outByteString
);
```

**Description**

Retrieves an element specified by an index number.

**Parameter**

---

arr	[In] The input byte-string array.
-----	-----------------------------------

---

index	[In] Specifies the zero-based index of the element.
-------	---

---

outByteString	[Out] It retrieves an element specified by an index number.
---------------	---

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1854

**FSByteStringArrayGetDataPtr****Syntax**

```
FS_ByteString FSByteStringArrayGetDataPtr (
    FS\_ByteStringArray arr,
    FS\_INT32 index
);
```

**Description**

The pointer to the specified element.

**Parameter**

---

arr	[In] The input byte-string array.
-----	-----------------------------------

---

---

index	[In] Specifies the zero-based index of the element.
-------	---

---

**Return**

Gets the pointer to the specified element.

**Head file reference**

fs\_basicTempl.h: 1914

**FSByteStringArrayGetSize****Syntax**

```
FS_INT32 FSByteStringArrayGetSize (
    FS\_ByteStringArray arr
);
```

**Description**

Gets the number of elements in the array.

**Parameter**

---

arr	[In] The input byte-string array.
-----	-----------------------------------

---

**Return**

The number of elements in the array.

**Head file reference**

fs\_basicTempl.h: 1865

**FSByteStringArrayNew****Syntax**

```
FS_ByteStringArray FSByteStringArrayNew (void );
```

**Description**

Creates a new empty byte-string array.

**Return**

A new empty byte-string array.

**Head file reference**

fs\_basicTempl.h: 1836

**FSByteStringArrayRemoveAll****Syntax**

```
void FSByteStringArrayRemoveAll (
```

```
FS_ByteStringArray arr  
);
```

**Description**

Cleans up the array.

**Parameter**

---

arr	[In] The input byte-string array.
-----	-----------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1874

**FSByteStringArrayRemoveAt****Syntax**

```
void FSByteStringArrayRemoveAt (  
    FS_ByteStringArray arr,  
    FS_INT32 index  
,
```

**Description**

Removes a number of elements at specified position.

**Parameter**

---

arr	[In] The input byte-string array.
-----	-----------------------------------

---

---

index	[In] Specifies the zero-based index in the array.
-------	---

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1894

## FS\_CharMap

[Return from Used by](#)

**Description**

A map struct for character mappings (encodings). All character maps should have the ability to convert between internal encoding and Unicode. See [FSCodeTransformationDecodeText](#) , [FSCodeTransformationDecodeText2](#) , [FSCodeTransformationEncodeText](#) .

### Returned from

[FPDFFontGetCharMap](#)  
[FSCharMapGetDefaultMapper](#)  
[FSCharMapGetDefaultMapper2](#)  
[FSCharMapNew](#)

### Used by

[FSByteStringConvertFrom](#)  
[FSCodeTransformationDecodeText](#)  
[FSCodeTransformationDecodeText2](#)  
[FSCodeTransformationEncodeText](#)  
[FSWideStringConvertFrom](#)  
[FSCharMapRelease](#)

## Functions

### Functions summary

#### [FSCharMapGetDefaultMapper](#)

Gets a character mapper according to Windows code page or other encoding system.

#### [FSCharMapGetDefaultMapper2](#)

Gets a character mapper according to Windows code page or other encoding system.

#### [FSCharMapNew](#)

Creates a new char mapper.

#### [FSCharMapRelease](#)

Destroys a character mapper.

### Functions detail

#### FSCharMapGetDefaultMapper

##### Syntax

```
FS_CharMap FSCharMapGetDefaultMapper (void );
```

##### Description

Gets a character mapper according to Windows code page or other encoding system. This char maps are managed by Foxit Reader, don't destroy them. This is system default mapper according to locale settings.

##### Return

A character mapper.

##### Head file reference

fs\_stringTempl.h: 28

**Related method**[FSCharMapNew](#)**FSCharMapGetDefaultMapper2****Syntax**

```
FS_CharMap FSCharMapGetDefaultMapper2 (
    FS\_INT32 codepage
);
```

**Description**

Gets a character mapper according to Windows code page or other encoding system. This char maps are managed by Foxit Reader, don't destroy them. This is system default mapper according to locale settings.

**Parameter**

---

codepage	[In] The input code page. Sets it 0 as default.
----------	---

---

**Return**

A character mapper.

**Head file reference**

fs\_stringTempl.h: 54

**Related method**[FSCharMapNew](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)**FSCharMapNew****Syntax**

```
FS_CharMap FSCharMapNew (
    FSCharmapGetWideString GetWideString,
    FSCharmapGetByteString GetByteString
);
```

**Description**

Creates a new char mapper.

**Parameter**

---

GetWideString	[In] A pointer type to <a href="#">FSCharmapGetWideString</a> .
---------------	---

---

---

GetByteString	[Out] A pointer type to <a href="#">FSCharmapGetByteString</a> .
---------------	--

---

**Return**

A newly created mapper

**Head file reference**

fs\_stringTempl.h: 21

**Related method**

[FSCharMapRelease](#)

[FSCharMapGetDefaultMapper](#)

**FSCharMapRelease****Syntax**

```
void FSCharMapRelease (
    FS\_CharMap mapper
);
```

**Description**

Destroys a character mapper.

**Parameter**

---

mapper	[In] The mapper to release.
--------	-----------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 27

**Related method**

[FSCharMapNew](#)

## FS\_CodeTransformation

### Description

Code transformation for name, string and text. See [FSCodeTransformationNameDecode](#) , [FSCodeTransformationNameDecode2](#) , [FSCodeTransformationNameEncode](#) , [FSCodeTransformationEncodeString](#) , [FSCodeTransformationDecodeText](#) , [FSCodeTransformationDecodeText2](#) , [FSCodeTransformationEncodeText](#) .

### Functions

#### Functions summary

**FSCodeTransformationBasicModuleA85Encode**

A85 encode algorithm. This interface is for internal use only now.

**FSCodeTransformationDecodeText**

Decode PDF encoded text string into a Unicode encoded string, from a byte string.

**FSCodeTransformationDecodeText2**

Decode PDF encoded text string into a Unicode encoded string, from a memory block.

**FSCodeTransformationEncodeString**

A PDF formatted string (including () or <>).

**FSCodeTransformationEncodeText**

Encode a Unicode text string (UTF-16LE) into PDF encoding.

**FSCodeTransformationFlateDecodeProc**

Flate decode algorithm.

**FSCodeTransformationFlateEncodeProc**

Flate encode algorithm.

**FSCodeTransformationFlateModuleEncode**

Flate module encode. This interface is for internal use only now.

**FSCodeTransformationFlateModuleEncode2**

Flate module encode. This interface is for internal use only now.

**FSCodeTransformationMemFree**

Free the pointer;

**FSCodeTransformationNameDecode**

Decode a name from its lexical form, From a buffered name of lexical form.

**FSCodeTransformationNameEncode**

Encode a name to lexical form (to be used for output).

**FSCodeTransformationRunLengthDecodeProc**

Run-length decode algorithm.

**FSCodeTransformationRunLengthEncodeProc**

Run-length encode algorithm.

## Functions detail

### FSCodeTransformationBasicModuleA85Encode

#### Syntax

```
FS_BOOL FSCodeTransformationBasicModuleA85Encode (
    FS_LPCBYTE src_buf,
    FS_DWORD src_size,
    FS_LPBYTE* dest_buf,
    FS_DWORD* dest_size
);
```

#### Description

A85 encode algorithm. This interface is for internal use only now.

#### Parameter

src_buf	[In] The source data.
---------	-----------------------

src_size	[In] The size in bytes of the source data.
----------	--

---

dest_buf	[Out] It will receive the encoded data. It must be released by <a href="#">FSCodeTransformationMemFree</a> .
----------	--

---

dest_size	[Out] It will receive the size in bytes of the encoded data.
-----------	--

---

**Return**

[TRUE](#) for success, [FALSE](#) for failure.

**Head file reference**

fs\_basicTempl.h: 2139

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FSCodeTransformationDecodeText****Syntax**

```
void FSCodeTransformationDecodeText (
    FS\_ByteString str,
    FS\_CharMap charMap,
    FS\_WideString* outDecodeText
);
```

**Description**

Decode PDF encoded text string into a Unicode encoded string, from a byte string.

**Parameter**


---

str	[In] The input PDF encoded text string.
-----	---

---

charMap	[In] The input character mapping.
---------	-----------------------------------

---

outDecodeText	[Out] A Unicode encoded string.
---------------	---------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2038

**Note:** If no char map specified, PDFDocEncoding is used.

**FSCodeTransformationDecodeText2****Syntax**

```
void FSCodeTransformationDecodeText2 (
    FS\_LPBYTE pData,
    FS\_DWORD size,
    FS\_CharMap charMap,
    FS\_WideString* outDecodeText
);
```

**Description**

Decode PDF encoded text string into a Unicode encoded string, from a memory block.

**Parameter**

---

pData	[In] The input PDF encoded text buffer.
size	[In] The size in bytes of the buffer.
charMap	[In] The input character mapping.
outDecodeText	[Out] A Unicode encoded string.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2049

**Note:** If no char map specified, PDFDocEncoding is used.

**FSCodeTransformationEncodeString****Syntax**

```
void FSCodeTransformationEncodeString (
    FS\_ByteString src,
    FS\_BOOL bHex,
    FS\_ByteString* outEncodedString
);
```

**Description**

A PDF formatted string (including () or <>).

**Parameter**

---

src	[In] The input string.
bHex	[In] Whether we will do hex-encoding.

---

---

outEncodedString	[Out] The PDF formatted string.
------------------	---------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2027

**FSCodeTransformationEncodeText****Syntax**

```
void FSCodeTransformationEncodeText (
    FS_LPWORD pString,
    FS_INT32 len,
    FS_CharMap charMap,
    FS_ByteString* outEncodeText
);
```

**Description**

Encode a Unicode text string (UTF-16LE) into PDF encoding.

**Parameter**


---

pString	[In] The input Unicode text string (UTF-16LE).
---------	--

---

len	[In] The length of the input unicode string. -1 for zero-terminated Unicode string.
-----	---

---

charMap	[In] The input character mapping.
---------	-----------------------------------

---

outEncodeText	[Out] The PDF encoded string.
---------------	-------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2061

**Note:** If no char map specified, PDFDocEncoding is used.

**FSCodeTransformationFlateDecodeProc****Syntax**

```
FS_DWORD FSCodeTransformationFlateDecodeProc (
    FS_LPCBYTE src_buf,
    FS_DWORD src_size,
```

---

```
FS_LPBYTE* dest_buf,
FS_DWORD* dest_size
);
```

**Description**

Flate decode algorithm.

**Parameter**


---

src_buf	[In] The source encoded data.
src_size	[In] The size in bytes of the source encoded data.
dest_buf	[Out] It will receive the decoded data. It must be released by <a href="#">FSCodeTransformationMemFree</a> .
dest_size	[Out] It will receive the size in bytes of the decoded data.

---

**Return**

The size in bytes of source data consumed.

**Head file reference**

fs\_basicTempl.h: 2082

**Related method**

[FSCodeTransformationFlateEncodeProc](#)

**Since**

[SDK LATEEST VERSION > 1.0](#)

### **FSCodeTransformationFlateEncodeProc**

**Syntax**

```
void FSCodeTransformationFlateEncodeProc (
    FS_LPCBYTE src_buf,
    FS_DWORD src_size,
    FS_LPBYTE* dest_buf,
    FS_DWORD* dest_size
);
```

**Description**

Flate encode algorithm.

**Parameter**


---

src_buf	[In] The source data.
---------	-----------------------

---

---

src_size	[In] The size in bytes of the source data.
dest_buf	[Out] It will receive the encoded data. It must be released by <a href="#">FSCodeTransformationMemFree</a> .
dest_size	[Out] It will receive the size in bytes of the encoded data.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2073

**Related method**[FSCodeTransformationFlateDecodeProc](#)**Since**[SDK LATEEST VERSION > 1.0](#)**FSCodeTransformationFlateModuleEncode****Syntax**

```
FS_BOOL FSCodeTransformationFlateModuleEncode (
    FS_LPCBYTE src_buf,
    FS_DWORD src_size,
    FS_LPBYTE* dest_buf,
    FS_DWORD* dest_size
);
```

**Description**

Flate module encode. This interface is for internal use only now.

**Parameter**


---

src_buf	[In] The source data.
src_size	[In] The size in bytes of the source data.
dest_buf	[Out] It will receive the encoded data. It must be released by <a href="#">FSCodeTransformationMemFree</a> .
dest_size	[Out] It will receive the size in bytes of the encoded data.

---

**Return**[TRUE](#) for success, [FALSE](#) for failure.

**Head file reference**

fs\_basicTempl.h: 2152

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FSCodeTransformationFlateModuleEncode2****Syntax**

```
FS_BOOL FSCodeTransformationFlateModuleEncode2 (
    FS_LPCBYTE src_buf,
    FS_DWORD src_size,
    FS_INT32 predictor,
    FS_INT32 Colors,
    FS_INT32 BitsPerComponent,
    FS_INT32 Columns,
    FS_LPBYTE* dest_buf,
    FS_DWORD* dest_size
);
```

**Description**

Flate module encode. This interface is for internal use only now.

**Parameter**


---

src_buf	[In] The source data.
---------	-----------------------

---

src_size	[In] The size in bytes of the source data.
----------	--

---

predictor	[In] The input predictor.
-----------	---------------------------

---

Colors	[In] The input colors.
--------	------------------------

---

BitsPerComponent	[In] The input bits per component.
------------------	------------------------------------

---

Columns	[In] The input columns.
---------	-------------------------

---

dest_buf	[Out] It will receive the encoded data. It must be released by <a href="#">FSCodeTransformationMemFree</a> .
----------	--

---

dest_size	[Out] It will receive the size in bytes of the encoded data.
-----------	--

---

**Return**[TRUE](#) for success, [FALSE](#) for failure.

**Head file reference**

fs\_basicTempl.h: 2165

**Since**[SDK\\_LATEEST\\_VERSION > 7.3.1](#)**FSCodeTransformationMemFree****Syntax**

```
void FSCodeTransformationMemFree (
    void* pointer
);
```

**Description**

Free the pointer;

**Parameter**

---

pointer	[In] The pointer to free;
---------	---------------------------

---

**Return**

void.

**Head file reference**

fs\_basicTempl.h: 2077

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FSCodeTransformationNameDecode****Syntax**

```
void FSCodeTransformationNameDecode (
    FS\_ByteString orig,
    FS\_ByteString* outDecodedName
);
```

**Description**

Decode a name from its lexical form, From a buffered name of lexical form.

**Parameter**

---

orig	[In] The input buffered name of lexical form.
------	---

---

outDecodedName	[Out] The decoded name.
----------------	-------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2007

**FSCodeTransformationNameEncode****Syntax**

```
void FSCodeTransformationNameEncode (
    FS\_ByteString orig,
    FS\_ByteString* outDecodedName
);
```

**Description**

Encode a name to lexical form (to be used for output).

**Parameter**

---

orig	[In] The input name.
------	----------------------

---

outDecodedName	[Out] A lexical form of the name.
----------------	-----------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2017

**FSCodeTransformationRunLengthDecodeProc****Syntax**

```
FS_DWORD FSCodeTransformationRunLengthDecodeProc (
    FS\_LPCBYTE src_buf,
    FS\_DWORD src_size,
    FS\_LPBYTE* dest_buf,
    FS\_DWORD* dest_size
);
```

**Description**

Run-length decode algorithm.

**Parameter**

---

src_buf	[In] The source encoded data.
---------	-------------------------------

---

src_size	[In] The size in bytes of the source encoded data.
----------	--

---

---

dest_buf	[Out] It will receive the decoded data. It must be released by <a href="#">FSCodeTransformationMemFree</a> .
----------	--

---

dest_size	[Out] It will receive the size in bytes of the decoded data.
-----------	--

---

**Return**

The size in bytes of source data consumed.

**Head file reference**

fs\_basicTempl.h: 2101

**Related method**

[FSCodeTransformationRunLengthEncodeProc](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FSCodeTransformationRunLengthEncodeProc****Syntax**

```
FS_BOOL FSCodeTransformationRunLengthEncodeProc (
    FS_LPCBYTE src_buf,
    FS_DWORD src_size,
    FS_LPBYTE* dest_buf,
    FS_DWORD* dest_size
);
```

**Description**

Run-length encode algorithm.

**Parameter**


---

src_buf	[In] The source data.
---------	-----------------------

---

src_size	[In] The size in bytes of the source data.
----------	--

---

dest_buf	[Out] It will receive the encoded data. It must be released by <a href="#">FSCodeTransformationMemFree</a> .
----------	--

---

dest_size	[Out] It will receive the size in bytes of the encoded data.
-----------	--

---

**Return****Head file reference**

fs\_basicTempl.h: 2110

**Related method**

[FSCodeTransformationRunLengthDecodeProc](#)**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)

## **FS\_DIBAttribute**

[Return from Used by](#)

### **Description**

An object representing a device-independent bitmap attribute. The bitmap has the attributes of horizontal resolution, vertical resolution, resolution unit and so on. See [FSDIBitmapLoadInfo](#).

### **Returned from**

[FSDIBitmapLoadInfo](#)

### **Used by**

[FSDIBitmapDestroyDIBAttribute](#)  
[FSDIBitmapGetDPIUnit](#)  
[FSDIBitmapGetExifInfo](#)  
[FSDIBitmapGetXDPI](#)  
[FSDIBitmapGetYDPI](#)

### **Definitions**

#### **Definitions summary**

**FS\_DIB\_EXIFTAG\_FLOAT\_DPIX**

Image resolution in width. The value type is 32 bits(FS\_FLOAT).

**FS\_DIB\_EXIFTAG\_FLOAT\_DPIY**

Image resolution in width. The value type is 32 bits(FS\_FLOAT).

**FS\_DIB\_EXIFTAG\_STRING\_COPYRIGHT**

Copyright holder. The value type is an 8-bit byte containing one 7-bit ASCII code. The final byte is terminated with NULL(FS\_LPSTR). The caller should not release the string.

**FS\_DIB\_EXIFTAG\_STRING\_DATETIME**

File change date and time. The value type is an 8-bit byte containing one 7-bit ASCII code. The final byte is terminated with NULL(FS\_LPSTR). The caller should not release the string.

**FS\_DIB\_EXIFTAG\_STRING\_IAMGEDESCRIPTION**

The description of bitmap. Don't support double byte characters yards, like Chinese/Japanese/Korean etc. The value type is an 8-bit byte containing one 7-bit ASCII code. The final byte is terminated with NULL(FS\_LPSTR). The caller should not release the string.

**FS\_DIB\_EXIFTAG\_STRING\_MANUFACTURER**

Digital camera manufacturers. The value type is an 8-bit byte containing one 7-bit ASCII code. The final byte is terminated with NULL(FS\_LPSTR). The caller should not release the string.

**FS\_DIB\_EXIFTAG\_STRING\_MODULE**

Digital camera module code. The value type is an 8-bit byte containing one 7-bit ASCII code. The final byte is terminated with NULL(FS\_LPSTR). The caller should not release the string.

**FS\_DIB\_EXIFTAG\_STRING\_SOFTWARE**

Software used. The value type is an 8-bit byte containing one 7-bit ASCII code. The final byte is terminated with NULL(FS\_LPSTR). The caller should not release the string.

**FS\_DIB\_EXIFTAG USHORT\_ORIENTATION**

The orientation. The value type is a 16-bit unsigned integer(FS\_WORD).

**FS\_DIB\_EXIFTAG USHORT\_RESUNIT**

Resolution unit. The value type is a 16-bit unsigned integer(FS\_WORD).

**Definitions detail****FS\_DIB\_EXIFTAG\_FLOAT\_DPIX****Syntax**

```
#define FS_DIB_EXIFTAG_FLOAT_DPIX 282
```

**Description**

Image resolution in width. The value type is 32 bits(FS\_FLOAT).

**Group**

[FSDIBEXIFTAG](#)

**Head file reference**

fs\_basicExpT.h: 1514

**FS\_DIB\_EXIFTAG\_FLOAT\_DPIY****Syntax**

```
#define FS_DIB_EXIFTAG_FLOAT_DPIY 283
```

**Description**

Image resolution in width. The value type is 32 bits(FS\_FLOAT).

**Group**

[FSDIBEXIFTAG](#)

**Head file reference**

fs\_basicExpT.h: 1516

**FS\_DIB\_EXIFTAG\_STRING\_COPYRIGHT****Syntax**

```
#define FS_DIB_EXIFTAG_STRING_COPYRIGHT 33432
```

**Description**

Copyright holder. The value type is an 8-bit byte containing one 7-bit ASCII code. The final byte is terminated with NULL(FS\_LPSTR). The caller should not release the string.

**Group**

[FSDIBEXIFTAG](#)

**Head file reference**

fs\_basicExpT.h: 1530

## FS\_DIB\_EXIFTAG\_STRING\_DATETIME

**Syntax**

#define FS\_DIB\_EXIFTAG\_STRING\_DATETIME 306

**Description**

File change date and time. The value type is an 8-bit byte containing one 7-bit ASCII code. The final byte is terminated with NULL(FS\_LPSTR). The caller should not release the string.

**Group**

[FSDIBEXIFTAG](#)

**Head file reference**

fs\_basicExpT.h: 1528

## FS\_DIB\_EXIFTAG\_STRING\_IAMGEDESCRIPTION

**Syntax**

#define FS\_DIB\_EXIFTAG\_STRING\_IAMGEDESCRIPTION 270

**Description**

The description of bitmap. Don't support double byte characters yards, like Chinese/Japanese/Korean etc. The value type is an 8-bit byte containing one 7-bit ASCII code. The final byte is terminated with NULL(FS\_LPSTR). The caller should not release the string.

**Group**

[FSDIBEXIFTAG](#)

**Head file reference**

fs\_basicExpT.h: 1526

## FS\_DIB\_EXIFTAG\_STRING\_MANUFACTURER

**Syntax**

#define FS\_DIB\_EXIFTAG\_STRING\_MANUFACTURER 271

**Description**

Digital camera manufacturers. The value type is an 8-bit byte containing one 7-bit ASCII code. The final byte is terminated with NULL(FS\_LPSTR). The caller should not release the string.

**Group**

[FSDIBEXIFTAG](#)

**Head file reference**

fs\_basicExpT.h: 1520

## FS\_DIB\_EXIFTAG\_STRING\_MODULE

**Syntax**

```
#define FS_DIB_EXIFTAG_STRING_MODULE 272
```

**Description**

Digital camera module code. The value type is an 8-bit byte containing one 7-bit ASCII code. The final byte is terminated with NULL(FS\_LPSTR). The caller should not release the string.

**Group**

[FSDIBEXIFTAG](#)

**Head file reference**

fs\_basicExpT.h: 1522

## FS\_DIB\_EXIFTAG\_STRING\_SOFTWARE

**Syntax**

```
#define FS_DIB_EXIFTAG_STRING_SOFTWARE 305
```

**Description**

Software used. The value type is an 8-bit byte containing one 7-bit ASCII code. The final byte is terminated with NULL(FS\_LPSTR). The caller should not release the string.

**Group**

[FSDIBEXIFTAG](#)

**Head file reference**

fs\_basicExpT.h: 1524

## FS\_DIB\_EXIFTAG\_USHORT\_ORIENTATION

**Syntax**

```
#define FS_DIB_EXIFTAG_USHORT_ORIENTATION 274
```

**Description**

The orientation. The value type is a 16-bit unsigned integer(FS\_WORD).

**Group**

[FSDIBEXIFTAG](#)

**Head file reference**

fs\_basicExpT.h: 1518

## FS\_DIB\_EXIFTAG USHORT RESUNIT

**Syntax**

```
#define FS_DIB_EXIFTAG USHORT RESUNIT 296
```

**Description**

Resolution unit. The value type is a 16-bit unsigned integer(FS\_WORD).

**Group**

[FSDIBEXIFTAG](#)

**Head file reference**

fs\_basicExpT.h: 1512

## Enumerations

### Enumerations summary

#### [FS\\_DIB\\_IMAGE\\_TYPE](#)

Device-independent bitmap image type.

#### [FS\\_DIB\\_RESUNIT](#)

Define the resolution unit type..

### Enumerations detail

#### FS\_DIB\_IMAGE\_TYPE

**Syntax**

```
enum FS_DIB_IMAGE_TYPE{  
    FS_DIB_IMAGE_UNKNOWN,  
    FS_DIB_IMAGE_BMP,  
    FS_DIB_IMAGE_JPG,  
    FS_DIB_IMAGE_PNG,  
    FS_DIB_IMAGE_GIF,  
    FS_DIB_IMAGE_TIF,  
    FS_DIB_IMAGE_MAX  
};
```

**Description**

Device-independent bitmap image type.

**Head file reference**

fs\_basicExpT.h: 1477

**FS\_DIB\_IMAGE\_UNKNOWN**

Unknown image, will be auto detected

**FS\_DIB\_IMAGE\_BMP**

Bmp

**FS\_DIB\_IMAGE\_JPG**

Jpeg

**FS\_DIB\_IMAGE\_PNG**

Png

**FS\_DIB\_IMAGE\_GIF**

Gif

**FS\_DIB\_IMAGE\_TIF**

Tiff

**FS\_DIB\_IMAGE\_MAX**

Max, the upperbound of image enum.

**FS\_DIB\_RESUNIT****Syntax**

```
enum FS_DIB_RESUNIT{
    FS_DIB_RESUNIT_NONE,
    FS_DIB_RESUNIT_INCH,
    FS_DIB_RESUNIT_CENTIMETER,
    FS_DIB_RESUNIT_METER
};
```

**Description**

Define the resolution unit type..

**Head file reference**

fs\_basicExpT.h: 1494

**FS\_DIB\_RESUNIT\_NONE**

no unit

**FS\_DIB\_RESUNIT\_INCH**

unit inch

**FS\_DIB\_RESUNIT\_CENTIMETER**

unit centimeter

**FS\_DIB\_RESUNIT\_METER**

unit meter

## FS\_DIBitmap

[Return from Used by](#)

### Description

An object representing a device-independent bitmap. DI Bitmap data are organized in [FS\\_DWORD](#) aligned scan lines, from top to bottom. Bitmap Coordinations: the bitmap coordinate has the origin at left-top corner. See [FSDIBitmapNew](#), [FSDIBitmapCreate](#), [FSDIBitmapDestroy](#).

### Returned from

[FRDocViewGetCurrentSnapshot](#)  
[FRMenuItemGetIcon](#)  
[FRWindowsDIBNew](#)  
[FRWindowsDIBLoadFromBuf](#)  
[FRWindowsDIBLoadFromDDB](#)  
[FRWindowsDIBLoadFromFile](#)  
[FRWindowsDIBLoadFromFileII](#)  
[FRWindowsDIBLoadDIBitmap](#)  
[FPDImageGetDIBSource](#)  
[FPDImageGetMask](#)  
[FPDImageLoadDIBitmap](#)  
[FPDRenderDeviceGetBitmap](#)  
[FPDTType3CharGetDIBitmap](#)  
[FSDIBitmapClone](#)  
[FSDIBitmapCloneConvert](#)  
[FSDIBitmapFlipImage](#)  
[FSDIBitmapGetAlphaMask](#)  
[FSDIBitmapLoadFromImage](#)  
[FSDIBitmapLoadFromPNGIcon](#)  
[FSDIBitmapLoadFromPNGIcon2](#)  
[FSDIBitmapLoadFromPNGIcon3](#)  
[FSDIBitmapNew](#)  
[FSDIBitmapStretchTo](#)  
[FSDIBitmapSwapXY](#)  
[FSDIBitmapTransformTo](#)

### Used by

[FRAppCollectBitmapData](#)  
[FRAppCreateCustomRecentFileList](#)  
[FRAppSetCustomRecentFileListItemBitmap](#)  
[FRAppSetRecentFileListImageByExt](#)  
[FRFuncBtnCreate](#)  
[FRFuncBtnUpdateImage](#)  
[FRMenuItemNew](#)  
[FRMenuItemSetIcon](#)  
[FRMessageBarAddButton](#)  
[FRMessageBarAddButton2](#)

[FRMessageBarAddButtonImage](#)  
[FRMessageBarAddButtonImage2](#)  
[FRMessageBarSetBitmap](#)  
[FRMessageBarSetBitmap2](#)  
[FRPanelMgrAddPanel](#)  
[FRPanelMgrAddPanel2](#)  
[FRRibbonBackStageViewItemAddPropertySheetPage](#)  
[FRRibbonBarAddButtonToAddPlace](#)  
[FRRibbonCategoryAddPanel](#)  
[FRRibbonCategoryAddPanel2](#)  
[FRRibbonElementAddChangeImage](#)  
[FRRibbonElementSetImage](#)  
[FRRibbonElementSetImplicitLargeBitmap](#)  
[FRRibbonElementSetTooltipImage](#)  
[FRRibbonListButtonAddGroup](#)  
[FRRibbonPaletteButtonAddGroup](#)  
[FRRibbonPaletteButtonInsertItemToGroupLast](#)  
[FRRibbonPaletteButtonSetDefaultGroup](#)  
[FRRibbonPanelSetPanellImage](#)  
[FRRibbonStyleButtonSetImage](#)  
[FRRibbonStyleButtonSetImage2](#)  
[FRRibbonStyleListBoxAddImage](#)  
[FRRibbonStyleListBoxAddImage2](#)  
[FRRibbonStyleStaticSetImage](#)  
[FRRibbonStyleStaticSetImage2](#)  
[FRToolButtonSetIcon](#)  
[FRWindowsDIBDestroy](#)  
[FRWindowsDIBGetBitmapInfo](#)  
[FRWindowsDIBGetDDBitmap](#)  
[FRWindowsDIBGetDC](#)  
[FRWindowsDIBGetWindowsBitmap](#)  
[FRWindowsDIBLoadFromDevice](#)  
[FRWindowsDIBSetToDevice](#)  
[FPDFxgeDeviceAttach](#)  
[FPDIImageLoadDIBitmap](#)  
[FPDIImageSetImage](#)  
[FPDPageRenderCacheGetCachedBitmap](#)  
[FPDRenderContextGetBackground](#)  
[FPDRenderDeviceCreateCompatibleBitmap](#)  
[FPDRenderDeviceGetDIBits](#)  
[FPDRenderDeviceSetBitmap](#)  
[FSDIBitmapClear](#)  
[FSDIBitmapClone](#)  
[FSDIBitmapCloneConvert](#)  
[FSDIBitmapCompositeBitmap](#)  
[FSDIBitmapCompositeRect](#)  
[FSDIBitmapConvertColorScale](#)  
[FSDIBitmapConvertFormat](#)  
[FSDIBitmapCopy](#)  
[FSDIBitmapCreate](#)  
[FSDIBitmapDestroy](#)  
[FSDIBitmapDitherFS](#)  
[FSDIBitmapDownSampleScanline](#)  
[FSDIBitmapFlipImage](#)  
[FSDIBitmapGammaAdjust](#)  
[FSDIBitmapGetAlphaMask](#)  
[FSDIBitmapGetBPP](#)

[FSDIBitmapGetBuffer](#)  
[FSDIBitmapGetFormat](#)  
[FSDIBitmapGetHeight](#)  
[FSDIBitmapGetPalette](#)  
[FSDIBitmapGetPaletteArgb](#)  
[FSDIBitmapGetPaletteSize](#)  
[FSDIBitmapGetPitch](#)  
[FSDIBitmapGetPixel](#)  
[FSDIBitmapGetScanline](#)  
[FSDIBitmapGetWidth](#)  
[FSDIBitmapHasAlpha](#)  
[FSDIBitmapIsAlphaMask](#)  
[FSDIBitmapIsOpaqueImage](#)  
[FSDIBitmapLoadChannel](#)  
[FSDIBitmapLoadChannel2](#)  
[FSDIBitmapMultiplyAlpha](#)  
[FSDIBitmapMultiplyAlpha2](#)  
[FSDIBitmapSetPaletteArgb](#)  
[FSDIBitmapSetPixel](#)  
[FSDIBitmapStretchTo](#)  
[FSDIBitmapSwapXY](#)  
[FSDIBitmapTakeOver](#)  
[FSDIBitmapTransferBitmap](#)  
[FSDIBitmapTransferMask](#)  
[FSDIBitmapTransformTo](#)

## Definitions

### Definitions summary

#### [FS\\_DIB\\_PALETTE\\_LOC](#)

When set, use the palette that built from a bitmap.

#### [FS\\_DIB\\_PALETTE\\_MAC](#)

When set, use the mac palette.

#### [FS\\_DIB\\_PALETTE\\_WIN](#)

When set, use the windows palette.

#### [FS\\_DIB\\_BLEND\\_COLOR](#)

Creates a color with the hue and saturation of the source color and the luminosity of the cdrop color.  $B(Cb, Cs) = \text{SetLum}(Cs, \text{Lum}(Cb))$

#### [FS\\_DIB\\_BLEND\\_COLORBURN](#)

Darkens the backdrop color to reflect the source color. Painting with white produces no changes.  $B(Cb, Cs) =$

- -  $1 - \min(1, (1 - Cb) / Cs)$  if  $Cs > 0$
- - 0 if  $Cs = 0$

#### [FS\\_DIB\\_BLEND\\_COLORDODGE](#)

Brightens the backdrop color to reflect the source color. Painting with black produces no changes.  $B(Cb, Cs) =$

- -  $\min(1, Cb / (1 - Cs))$  if  $Cs < 1$
- - 1 if  $Cs = 1$

#### [FS\\_DIB\\_BLEND\\_DARKEN](#)

Selects the darker of the backdrop and source colors.  $B(Cb, Cs) = \min(Cb, Cs)$

#### **FS\_DIB\_BLEND\_DIFFERENCE**

Subtracts the darker of the two constituent colors from the lighter color.  $B(Cb, Cs) = |Cb - Cs|$

#### **FS\_DIB\_BLEND\_EXCLUSION**

Produces an effect similar to that of the difference mode but lower in contrast.  $B(Cb, Cs) = Cb + Cs - 2 * Cb * Cs$

#### **FS\_DIB\_BLEND\_HARDLIGHT**

Multiplies or screens the colors, depending on the source color value.  $B(Cb, Cs) =$

- - Multiply( $Cb, 2 * Cs$ ) if  $Cs \leq 0.5$
- - Screen( $Cb, 2 * Cs - 1$ ) if  $Cs > 0.5$

#### **FS\_DIB\_BLEND\_HUE**

Creates a color with the hue of the source color and the saturation and luminosity of the backdrop color.  $B(Cb, Cs) = \text{SetLum}(\text{SetSat}(Cs, \text{Sat}(Cb)), \text{Lum}(Cb))$

#### **FS\_DIB\_BLEND\_LIGHTEN**

Selects the lighter of the backdrop and source colors.  $B(Cb, Cs) = \max(Cb, Cs)$

#### **FS\_DIB\_BLEND\_LUMINOSITY**

Creates a color with the luminosity of the source color and the hue and saturation of the backdrop color.  $B(Cb, Cs) = \text{SetLum}(Cb, \text{Lum}(Cs))$

#### **FS\_DIB\_BLEND\_MULTIPLY**

Multiplies the backdrop and source color values.  $B(Cb, Cs) = Cb * Cs$

#### **FS\_DIB\_BLEND\_NONSEPARABLE**

Standard nonseparable blend modes *SetLum* , *Lum* , *SetSat* , *Sat* defined in the p.524 of the pdf reference 1.7.

#### **FS\_DIB\_BLEND\_NORMAL**

Selects the source color, ignoring the backdrop.  $B(Cb, Cs) = Cs$

#### **FS\_DIB\_BLEND\_OVERLAY**

Multiplies or screens the colors, depending on the backdrop color value.  $B(Cb, Cs) = \text{HardLight}(Cs, Cb)$

#### **FS\_DIB\_BLEND\_SATURATION**

Creates a color with the saturation of the source color and the hue and luminosity of the backdrop color.  $B(Cb, Cs) = \text{SetLum}(\text{SetSat}(Cb, \text{Sat}(Cs)), \text{Lum}(Cb))$

#### **FS\_DIB\_BLEND\_SCREEN**

Multiplies the complements of the backdrop and source color values, then complements the result.  $B(Cb, Cs) = 1 - [(1 - Cb) * (1 - Cs)] = Cb + Cs - Cb * Cs$

#### **FS\_DIB\_BLEND\_SOFTLIGHT**

Darkens or lightens the colors, depending on the source color value.  $B(Cb, Cs) =$

- -  $Cb - (1 - 2 * Cs) * Cb * (1 - Cb)$  if  $Cs \leq 0.5$
- -  $Cb + (2 * Cs - 1) * (D(Cb) - Cb)$  if  $Cs > 0.5$  \n

where  $D(x) =$

- -  $((16 * x - 12) * x + 4) * x$  if  $x \leq 0.25$
- -  $\sqrt{x}$  if  $x > 0.25$

#### **FS\_DIB\_BLEND\_UNSUPPORTED**

Unsupported blend type.

#### **FS\_DIB\_DOWNSAMPLE**

When set, don't do halftone for shrinking or rotating.

**FS\_DIB\_INTERPOL**

When set, do interpolation for enlarging.

**FSFILL\_ALTERNATE**

Alternate.

**FSFILL\_FULLCOVER**

A special flag that can be applied to fill mode, indicating all pixels partially covered by the path will be fully painted.

**FSFILL\_RECT\_AA**

A special flag that can be applied to fill mode, indicating the rectangle won't be fully filled (like by default), instead, it will be filled using standard anti-aliasing.

**FSFILL\_WINDING**

Winding.

**Definitions detail****FS\_DIB\_PALETTE\_LOC****Syntax**

```
#define FS_DIB_PALETTE_LOC 0x01
```

**Description**

When set, use the palette that built from a bitmap.

**Group**

[FSDIBConvertFlags](#)

**Head file reference**

fs\_basicExpT.h: 636

**FS\_DIB\_PALETTE\_MAC****Syntax**

```
#define FS_DIB_PALETTE_MAC 0x04
```

**Description**

When set, use the mac palette.

**Group**

[FSDIBConvertFlags](#)

**Head file reference**

fs\_basicExpT.h: 640

**FS\_DIB\_PALETTE\_WIN****Syntax**

```
#define FS_DIB_PALETTE_WIN 0x02
```

**Description**

When set, use the windows palette.

**Group**

[FSDIBConvertFlags](#)

**Head file reference**

fs\_basicExpT.h: 638

## FS\_DIB\_BLEND\_COLOR

**Syntax**

#define FS\_DIB\_BLEND\_COLOR 23

**Description**

Creates a color with the hue and saturation of the source color and the luminosity of the cdkdrop color.  $B(Cb, Cs) = \text{SetLum}(Cs, \text{Lum}(Cb))$

**Group**

[FSDIBBlendTypes](#)

**Head file reference**

fs\_basicExpT.h: 791

## FS\_DIB\_BLEND\_COLORBURN

**Syntax**

#define FS\_DIB\_BLEND\_COLORBURN 7

**Description**

Darkens the backdrop color to reflect the source color. Painting with white produces no changes.  $B(Cb, Cs) =$

- - 1 -  $\min(1, (1 - Cb) / Cs)$  if  $Cs > 0$
- - 0 if  $Cs = 0$

**Group**

[FSDIBBlendTypes](#)

**Head file reference**

fs\_basicExpT.h: 732

## FS\_DIB\_BLEND\_COLORDODGE

**Syntax**

#define FS\_DIB\_BLEND\_COLORDODGE 6

**Description**

Brightens the backdrop color to reflect the source color. Painting with black produces no changes.  $B(C_b, C_s) =$

- -  $\min(1, C_b / (1 - C_s))$  if  $C_s < 1$
- - 1 if  $C_s = 1$

**Group**

[FSDIBlendTypes](#)

**Head file reference**

fs\_basicExpT.h: 723

## FS\_DIB\_BLEND\_DARKEN

**Syntax**

#define FS\_DIB\_BLEND\_DARKEN 4

**Description**

Selects the darker of the backdrop and source colors.  $B(C_b, C_s) = \min(C_b, C_s)$

**Group**

[FSDIBlendTypes](#)

**Head file reference**

fs\_basicExpT.h: 709

## FS\_DIB\_BLEND\_DIFFERENCE

**Syntax**

#define FS\_DIB\_BLEND\_DIFFERENCE 10

**Description**

Subtracts the darker of the two constituent colors from the lighter color.  $B(C_b, C_s) = |C_b - C_s|$

**Group**

[FSDIBlendTypes](#)

**Head file reference**

fs\_basicExpT.h: 760

## FS\_DIB\_BLEND\_EXCLUSION

**Syntax**

#define FS\_DIB\_BLEND\_EXCLUSION 11

**Description**

Produces an effect similar to that of the difference mode but lower in contrast.  $B(Cb, Cs) = Cb + Cs - 2 * Cb * Cs$

**Group**

[FSDIBlendTypes](#)

**Head file reference**

fs\_basicExpT.h: 765

## FS\_DIB\_BLEND\_HARDLIGHT

**Syntax**

#define FS\_DIB\_BLEND\_HARDLIGHT 8

**Description**

Multiples or screens the colors, depending on the source color value.  $B(Cb, Cs) =$

- - Multiply( $Cb, 2 * Cs$ ) if  $Cs \leq 0.5$
- - Screen( $Cb, 2 * Cs - 1$ ) if  $Cs > 0.5$

**Group**

[FSDIBlendTypes](#)

**Head file reference**

fs\_basicExpT.h: 741

## FS\_DIB\_BLEND\_HUE

**Syntax**

#define FS\_DIB\_BLEND\_HUE 21

**Description**

Creates a color with the hue of the source color and the saturation and luminosity of the backdrop color.  $B(Cb, Cs) = \text{SetLum}(\text{SetSat}(\text{Cs}, \text{Sat}(Cb)), \text{Lum}(Cb))$

**Group**

[FSDIBlendTypes](#)

**Head file reference**

fs\_basicExpT.h: 777

## FS\_DIB\_BLEND\_LIGHTEN

**Syntax**

#define FS\_DIB\_BLEND\_LIGHTEN 5

**Description**

Selects the lighter of the backdrop and source colors.  $B(Cb, Cs) = \max(Cb, Cs)$

**Group**

[FSDIBlendTypes](#)

**Head file reference**

fs\_basicExpT.h: 714

## FS\_DIB\_BLEND\_LUMINOSITY

**Syntax**

```
#define FS_DIB_BLEND_LUMINOSITY 24
```

**Description**

Creates a color with the luminosity of the source color and the hue and saturation of the backdrop color.  $B(Cb, Cs) = \text{SetLum}(Cb, \text{Lum}(Cs))$

**Group**

[FSDIBlendTypes](#)

**Head file reference**

fs\_basicExpT.h: 798

## FS\_DIB\_BLEND\_MULTIPLY

**Syntax**

```
#define FS_DIB_BLEND_MULTIPLY 1
```

**Description**

Multiplies the backdrop and source color values.  $B(Cb, Cs) = Cb * Cs$

**Group**

[FSDIBlendTypes](#)

**Head file reference**

fs\_basicExpT.h: 694

## FS\_DIB\_BLEND\_NONSEPARABLE

**Syntax**

```
#define FS_DIB_BLEND_NONSEPARABLE 21
```

**Description**

Standard nonseparable blend modes *SetLum* , *Lum* , *SetSat* , *Sat* defined in the p.524 of the pdf reference 1.7.

**Group**

[FSDIBlendTypes](#)

**Head file reference**

fs\_basicExpT.h: 770

**FS\_DIB\_BLEND\_NORMAL****Syntax**

#define FS\_DIB\_BLEND\_NORMAL 0

**Description**Selects the source color, ignoring the backdrop.  $B(C_b, C_s) = C_s$ 

- - B denotes blend function. It can be Normal, Multiply, Screen...
- - Cr denotes result color components.
- - Cb denotes background color components.
- - Cs denotes source color components.

See [FSDIBitmapCompositeBitmap](#) , [FSDIBitmapCompositeMask](#) .**Group**[FSDIBlendTypes](#)**Head file reference**

fs\_basicExpT.h: 689

**FS\_DIB\_BLEND\_OVERLAY****Syntax**

#define FS\_DIB\_BLEND\_OVERLAY 3

**Description**Multiplies or screens the colors, depending on the backdrop color value.  $B(C_b, C_s) = \text{HardLight}(C_s, C_b)$ **Group**[FSDIBlendTypes](#)**Head file reference**

fs\_basicExpT.h: 704

**FS\_DIB\_BLEND\_SATURATION****Syntax**

#define FS\_DIB\_BLEND\_SATURATION 22

**Description**Creates a color with the saturation of the source color and the hue and luminosity of the backdrop color.  $B(C_b, C_s) = \text{SetLum}(\text{SetSat}(C_b, \text{Sat}(C_s)), \text{Lum}(C_b))$

**Group**[FSDIBlendTypes](#)**Head file reference**

fs\_basicExpT.h: 784

**FS\_DIB\_BLEND\_SCREEN****Syntax**

#define FS\_DIB\_BLEND\_SCREEN 2

**Description**

Multiplies the complements of the backdrop and source color values, then complements the result.  $B(Cb, Cs) = 1 - [(1 - Cb) * (1 - Cs)] = Cb + Cs - Cb * Cs$

**Group**[FSDIBlendTypes](#)**Head file reference**

fs\_basicExpT.h: 699

**FS\_DIB\_BLEND\_SOFTLIGHT****Syntax**

#define FS\_DIB\_BLEND\_SOFTLIGHT 9

**Description**

Darkens or lightens the colors, depending on the source color value.  $B(Cb, Cs) =$

- $- Cb - (1 - 2 * Cs) * Cb * (1 - Cb)$  if  $Cs \leq 0.5$
- $- Cb + (2 * Cs - 1) * (D(Cb) - Cb)$  if  $Cs > 0.5$  \n

where  $D(x) =$

- $- ((16 * x - 12) * x + 4) * x$  if  $x \leq 0.25$
- $- \sqrt{x}$  if  $x > 0.25$

**Group**[FSDIBlendTypes](#)**Head file reference**

fs\_basicExpT.h: 755

**FS\_DIB\_BLEND\_UNSUPPORTED**

**Syntax**

```
#define FS_DIB_BLEND_UNSUPPORTED -1
```

**Description**

Unsupported blend type.

**Group**

[FSDIBlendTypes](#)

**Head file reference**

fs\_basicExpT.h: 801

## FS\_DIB\_DOWNSAMPLE

**Syntax**

```
#define FS_DIB_DOWNSAMPLE 0x04
```

**Description**

When set, don't do halftone for shrinking or rotating.

**Group**

[FSDIBStretchFlags](#)

**Head file reference**

fs\_basicExpT.h: 624

## FS\_DIB\_INTERPOL

**Syntax**

```
#define FS_DIB_INTERPOL 0x20
```

**Description**

When set, do interpolation for enlarging.

**Group**

[FSDIBStretchFlags](#)

**Head file reference**

fs\_basicExpT.h: 626

## FSFILL\_ALTERNATE

**Syntax**

```
#define FSFILL_ALTERNATE 1
```

**Description**

Alternate.

**Group**

[FSFillingModeFlags](#)

**Head file reference**

fs\_basicExpT.h: 651

## FSFILL\_FULLSCREEN

**Syntax**

#define FSFILL\_FULLSCREEN 4

**Description**

A special flag that can be applied to fill mode, indicating ll pixels partially covered by the path will be fully painted.

**Group**

[FSFillingModeFlags](#)

**Head file reference**

fs\_basicExpT.h: 658

## FSFILL\_RECT\_AA

**Syntax**

#define FSFILL\_RECT\_AA 8

**Description**

A special flag that can be applied to fill mode, indicating the rectangle won't be fully filled (like by default), instead, it will filled using standard anti-aliasing.

**Group**

[FSFillingModeFlags](#)

**Head file reference**

fs\_basicExpT.h: 664

## FSFILL\_WINDING

**Syntax**

#define FSFILL\_WINDING 2

**Description**

Winding.

**Group**

[FSFillingModeFlags](#)

**Head file reference**

fs\_basicExpT.h: 653

## Enumerations

### Enumerations summary

**FS\_DIB\_Channel**

Device-independent bitmap channel ID. See [FSDIBitmapLoadChannel](#) , [FSDIBitmapLoadChannel2](#) .

**FS\_DIB\_Format**

Device-independent bitmap formats. See [FSDIBitmapConvertFormat](#) .

### Enumerations detail

**FS\_DIB\_Channel****Syntax**

```
enum FS_DIB_Channel{  
    FS\_DIB\_Red,  
    FS\_DIB\_Green,  
    FS\_DIB\_Blue,  
    FS\_DIB\_Alpha  
};
```

**Description**

Device-independent bitmap channel ID. See [FSDIBitmapLoadChannel](#) , [FSDIBitmapLoadChannel2](#) .

**Head file reference**

fs\_basicExpT.h: 606

**FS\_DIB\_Red**

Red channel.

**FS\_DIB\_Green**

Green channel.

**FS\_DIB\_Blue**

Blue channel.

**FS\_DIB\_Alpha**

Alpha channel.

**FS\_DIB\_Format****Syntax**

```
enum FS_DIB_Format{  
    FS\_DIB\_Invalid,  
    FS\_DIB\_1bppMask,
```

```
    FS\_DIB\_1bppRgb,  
    FS\_DIB\_8bppMask,  
    FS\_DIB\_8bppRgb,  
    FS\_DIB\_Rgb,  
    FS\_DIB\_Rgb32,  
    FS\_DIB\_Argb  
};
```

### Description

Device-independent bitmap formats. See [FSDIBitmapConvertFormat](#).

### Head file reference

fs\_basicExpT.h: 589

#### **FS\_DIB\_Invalid**

not a real format.

#### **FS\_DIB\_1bppMask**

1bpp bit mask. 0 for zero alpha, 1 for maximum alpha.

#### **FS\_DIB\_1bppRgb**

1bpp two color RGB bitmap.

#### **FS\_DIB\_8bppMask**

8bpp alpha mask.

#### **FS\_DIB\_8bppRgb**

8bpp 256 color RGB bitmap.

#### **FS\_DIB\_Rgb**

24bpp RGB bitmap. Byte order: BGR.

#### **FS\_DIB\_Rgb32**

32bpp RGB bitmap. Byte order: BGR, 4th byte not used.

#### **FS\_DIB\_Argb**

32bpp ARGB bitmap. Byte order: BGRA.

## Typedefs

### Typedefs summary

#### **[FS\\_ARGB](#)**

ARGB color type.

#### **[FS\\_COLORREF](#)**

RGB color type.

### Typedefs detail

#### **FS\_ARGB**

**Syntax**

```
typedef FS_DWORD FS_ARGB;
```

**Description**

ARGB color type.

**Head file reference**

fs\_basicExpT.h: 809

**FS\_COLORREF****Syntax**

```
typedef FS_DWORD FS_COLORREF;
```

**Description**

RGB color type.

**Head file reference**

fs\_basicExpT.h: 811

## Functions

### Functions summary

**[FSDIBitmapClear](#)**

Fills the whole bitmap with specified color. For alpha mask bitmaps, only the alpha value is taken.

**[FSDIBitmapClone](#)**

Clones a bitmap. The returned bitmap must be destroyed by [FSDIBitmapDestroy](#) () .

**[FSDIBitmapCloneConvert](#)**

Converts a bitmap, but clone first. The returned bitmap must be destroyed by [FSDIBitmapDestroy](#) () .

**[FSDIBitmapCompositeBitmap](#)**

Composites and blend source bitmap into this *bitmap* .

This bitmap can be Rgb/Rgb32/Argb bitmap, and source bitmap can be any kind of bitmap (except masks).

This bitmap can also be an 8bppMask, in this case only alpha channel from the source bitmap is composited.

**[FSDIBitmapCompositeRect](#)**

Composites a fixed color into a rectangle area.

**[FSDIBitmapConvertColorScale](#)**

Converts current bitmap to a color scale bitmap. The DIB format won't be changed.

**[FSDIBitmapConvertFormat](#)**

Converts a bitmap. All informations in the old bitmap are retained.

**[FSDIBitmapCopy](#)**

Copy from a DIB source, including bitmap info and all pixel data. This DIBitmap must be newly constructed.

**[FSDIBitmapCreate](#)**

Actually create the *DIB* . Optionally the *DIB* can use external buffer provided by caller.

**[FSDIBitmapDestroy](#)**

Destroys the bitmap.

**[FSDIBitmapDestroyDIBAttribute](#)**

Destroys the input bitmap attribute object. Otherwise the plug-in will cause the memory leak problem.

**[FSDIBitmapDitherFS](#)**

Floyd-Steinberg dithering for (portion of) the bitmap, using a palette.

**[FSDIBitmapDownSampleScanline](#)**

Downs sample a scanline, for quick stretching.

**[FSDIBitmapFlipImage](#)**

Flips image. The returned bitmap must be destroyed by [FSDIBitmapDestroy](#) () .

**[FSDIBitmapGammaAdjust](#)**

Gamma adjustment.

**[FSDIBitmapGetAlphaMask](#)**

Gets alpha mask from a bitmap. Returns a 8bpp alpha mask. The returned value must be destroyed by [FSDIBitmapDestroy](#) () .

**[FSDIBitmapGetBPP](#)**

Gets the number of bits per pixel.

**[FSDIBitmapGetBuffer](#)**

Gets a buffer for whole *DIB* . Only in-memory DIB can supply such a buffer.

**[FSDIBitmapGetDPIUnit](#)**

Gets the resolution unit, described by [FS\\_DIB\\_RESUNIT](#) .

**[FSDIBitmapGetExifInfo](#)**

Gets the exchangeable image file information of camera in JPEG file.

**[FSDIBitmapGetFormat](#)**

Gets the format of the bitmap.

**[FSDIBitmapGetHeight](#)**

Gets the height of the bitmap.

**[FSDIBitmapGetPalette](#)**

Gets the palette of the bitmap.

**[FSDIBitmapGetPaletteArgb](#)**

Gets palette entry with specified palette entry index.

**[FSDIBitmapGetPaletteSize](#)**

Gets the number of palette entries.

**[FSDIBitmapGetPitch](#)**

Gets the specified row pitch of the bitmap.

**[FSDIBitmapGetPixel](#)**

Gets pixel ARGB value.

**[FSDIBitmapGetScanline](#)**

Fetches a single scan line.

**[FSDIBitmapGetWidth](#)**

Gets the width of the bitmap.

**[FSDIBitmapGetXDPI](#)**

Gets the horizontal resolution attribute of the bitmap.

**[FSDIBitmapGetYDPI](#)**

Gets the Vertical resolution attribute of the bitmap.

**[FSDIBitmapHasAlpha](#)**

Checks if it's a bitmap with alpha channel.

**[FSDIBitmapIsAlphaMask](#)**

Checks whether the bitmap is an alpha mask (either 1bpp bitmask or 8bpp gray scale).

**[FSDIBitmapIsOpaqueImage](#)**

Checks if it's a solid (opaque) image.

**[FSDIBitmapLoadChannel](#)**

Loads a full channel from a source bitmap to this bitmap.

**[FSDIBitmapLoadChannel2](#)**

Sets a full channel to specified value (0-255).

**[FSDIBitmapLoadFromImage](#)**

Loads a bitmap from a buffer.

**[FSDIBitmapLoadFromPNGIcon](#)**

Loads a bitmap from a PNG file.

**[FSDIBitmapLoadFromPNGIcon2](#)**

Loads a bitmap from a buffer.

**[FSDIBitmapLoadFromPNGIcon3](#)**

Loads a bitmap from a instance handle.

**[FSDIBitmapLoadInfo](#)**

Loads a bitmap attribute from a buffer.

**[FSDIBitmapMultiplyAlpha](#)**

Multiplies alpha data with addition alpha (0-255).

**[FSDIBitmapMultiplyAlpha2](#)**

Multiplies existing alpha data with another alpha mask.

**[FSDIBitmapNew](#)**

Creates a bitmap. Optionally an external buffer provided by caller can be used. Using [FSDIBitmapCreate](#) () to initialize after a bitmap object is created.

**[FSDIBitmapSetPaletteArgb](#)**

Changes a specified palette entry.

**[FSDIBitmapSetPixel](#)**

Sets pixel ARGB value.

**[FSDIBitmapStretchTo](#)**

Stretches this bitmap into a new bitmap with different size. The returned bitmap must be destroyed by [FSDIBitmapDestroy](#) ().

**[FSDIBitmapSwapXY](#)**

Swaps X,Y coordinations of the bitmap. The image can also be flipped at the same time. The returned bitmap must be destroyed by [FSDIBitmapDestroy](#) ().

**[FSDIBitmapTakeOver](#)**

Takeovers a bitmap.

**[FSDIBitmapTransferBitmap](#)**

Transforms pixels from another bitmap into specified position.

**[FSDIBitmapTransferMask](#)**

Transfers (portion of) an alpha mask (with source color) to this bitmap.

**[FSDIBitmapTransformTo](#)**

Transforms this bitmap. A new transformed bitmap is returned. The returned bitmap must be destroyed by [FSDIBitmapDestroy](#) ().

## Functions detail

### **FSDIBitmapClear**

#### **Syntax**

```
void FSDIBitmapClear (
    FS_DIBitmap bitmap,
    FS_ARGB argb
);
```

**Description**

Fills the whole bitmap with specified color. For alpha mask bitmaps, only the alpha value is taken.

**Parameter**

---

bitmap	[In] The input bitmap.
--------	------------------------

---

argb	[In] The specified color to fill.
------	-----------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 286

**FSDIBitmapClone****Syntax**

```
FS_DIBitmap FSDIBitmapClone (
    FS\_DIBitmap bitmap,
    FS\_Rect* pClip
);
```

**Description**

Clones a bitmap. The returned bitmap must be destroyed by [FSDIBitmapDestroy](#) () .

**Parameter**

---

bitmap	[In] The input bitmap.
--------	------------------------

---

pClip	[In] The clipping region of source bitmap
-------	---

---

**Return**

A cloned bitmap.

**Head file reference**

fs\_basicTempl.h: 713

**Related method**

[FSDIBitmapCloneConvert](#)

**Note:** Optionally a clipping region in bitmap coordinates can be specified to limit the size of result bitmap.

## FSDIBitmapCloneConvert

### Syntax

```
FS_DIBitmap FSDIBitmapCloneConvert (
    FS_DIBitmap bitmap,
    FS_DIB_Format format,
    FS_Rect* pClip
);
```

### Description

Converts a bitmap, but clone first. The returned bitmap must be destroyed by [FSDIBitmapDestroy \(\)](#).

### Parameter

bitmap	[In] The input bitmap.
format	[In] The destination bitmap format.
pClip	[In] The clipping region of source bitmap.

### Return

The cloned bitmap.

### Head file reference

fs\_basicTempl.h: 719

### Related method

[FSDIBitmapClone](#)

**Note:** Optionally a clipping region in bitmap coordinates can be specified to limit the size of result bitmap.

## FSDIBitmapCompositeBitmap

### Syntax

```
void FSDIBitmapCompositeBitmap (
    FS_DIBitmap bitmap,
    FS_INT32 destLeft,
    FS_INT32 destTop,
    FS_INT32 width,
    FS_INT32 height,
    const FS_DIBitmap srcBitmap,
    srcLeft,
    srcTop,
    FS_INT32 blend_type
);
```

**Description**

Composites and blend source bitmap into this *bitmap*.  
 This bitmap can be Rgb/Rgb32/Argb bitmap, and source bitmap can be any kind of bitmap (except masks).  
 This bitmap can also be an 8bppMask, in this case only alpha channel from the source bitmap is composited.

**Parameter**


---

bitmap	[In] The input bitmap.
destLeft	[In] The x-coordinate in destination bitmap.
destTop	[In] The y-coordinate in destination bitmap.
width	[In] The area width to composite.
height	[In] The area height to composite.
srcBitmap	[In] The source bitmap.
srcLeft	[In] The x-coordinate in source bitmap.
srcTop	[In] The y-coordinate in source bitmap.
blend_type	[In] The blend type. Declared in <a href="#">FSDIBlendTypes</a> group.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 565

**Related method**

[FSDIBitmapCompositeMask](#)  
[FSDIBitmapCompositeRect](#)

**FSDIBitmapCompositeRect****Syntax**

```
void FSDIBitmapCompositeRect (
  FS\_DIBitmap bitmap,
  FS\_INT32 destLeft,
  FS\_INT32 destTop,
```

---

```
FS_INT32 width,  
FS_INT32 height,  
FS_ARGB argb  
);
```

**Description**

Composites a fixed color into a rectangle area.

**Parameter**

bitmap	[In] The input bitmap.
destLeft	[In] The x-coordinate of the left-top corner.
destTop	[In] The y-coordinate of the left-top corner.
width	[In] The area width.
height	[In] The area height.
argb	[In] The fixed color.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 567

**Related method**

[FSDIBitmapCompositeBitmap](#)  
[FSDIBitmapTransferMask](#)  
[FSDIBitmapCompositeMask](#)

**Note:** Applicable to 8bppMask/Rgb/Rgb32/Argb formats only.

**FSDIBitmapConvertColorScale****Syntax**

```
void FSDIBitmapConvertColorScale (  
    FS_DIBitmap bitmap,  
    FS_COLORREF forecolor,  
    FS_COLORREF backcolor  
);
```

**Description**

Converts current bitmap to a color scale bitmap. The DIB format won't be changed.

**Parameter**

---

bitmap	[In] The input bitmap.
forecolor	[In] The input forecolor.
backcolor	[In] The input backcolor.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 620

**Note:** Color scale means all colors are a scale from forecolor to backcolor. If forecolor is black and backcolor is white, that's gray scale Doesn't work with masks.

**FSDIBitmapConvertFormat****Syntax**

```
void FSDIBitmapConvertFormat (
    FS\_DIBitmap bitmap,
    FS\_DIB\_Format format
);
```

**Description**

Converts a bitmap. All informations in the old bitmap are retained.

**Parameter**

---

bitmap	[In] The input bitmap.
format	[In] The destination bitmap format.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 416

**Note:** Supported conversion:

- 1bppMask => 8bppMask;
- 
- 1bppRgb => Rgb / Rgb32 / Argb;
- 
- 8bppRgb => Rgb / Rgb32 / Argb;
- 
- Rgb / Rgb32 => 8bppRgb;
- 
- Rgb => Rgb32 / Argb;
- 
- Rgb32 => Rgb / Argb

## FSDIBitmapCopy

### Syntax

```
FS_BOOL FSDIBitmapCopy (
    FS_DIBitmap bitmap,
    FS_DIBitmap src
);
```

### Description

Copy from a DIB source, including bitmap info and all pixel data. This DIBitmap must be newly constructed.

### Parameter

---

bitmap	[In] The input bitmap.
--------	------------------------

---

src	[In] The DIB source.
-----	----------------------

---

### Return

TRUE for success, FALSE for failure.

### Head file reference

fs\_basicTempl.h: 866

### Since

[SDK LATEEST VERSION > 1.0](#)

**FSDIBitmapCreate****Syntax**

```
void FSDIBitmapCreate (
    FS\_DIBitmap bitmap,
    FS\_INT32 width,
    FS\_INT32 height,
    FS\_DIB\_Format format,
    FS\_LPBYTE pBuffer,
    FS\_INT32 pitch
);
```

**Description**

Actually create the *DIB* . Optionally the *DIB* can use external buffer provided by caller.

**Parameter**

bitmap	[In] The input bitmap.
width	[In] The width of the bitmap.
height	[In] The height of the bitmap.
format	[In] The format of the bitmap.
pBuffer	[In] The data buffer of the bitmap.
pitch	[In] Specified row pitch in bytes.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 283

**Related method**

[FSDIBitmapNew](#)

[FSDIBitmapClear](#)

[FSDIBitmapDestroy](#)

**Note:**The buffer should be kept by caller during the existence of the bitmap.

**FSDIBitmapDestroy**

**Syntax**

```
void FSDIBitmapDestroy (
    FS\_DIBitmap bitmap
);
```

**Description**

Destroys the bitmap.

**Parameter**

---

bitmap	[In] The input bitmap.
--------	------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 288

**Related method**

[FSDIBitmapCreate](#)

[FSDIBitmapNew](#)

**FSDIBitmapDestroyDIBAttribute****Syntax**

```
void FSDIBitmapDestroyDIBAttribute (
    FS\_DIBAttribute attr
);
```

**Description**

Destroys the input bitmap attribute object. Otherwise the plug-in will cause the memory leak problem.

**Parameter**

---

attr	[In] The input bitmap attribute object to be destroyed.
------	---

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 885

**Related method**

[FSDIBitmapLoadInfo](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FSDIBitmapDitherFS

### Syntax

```
void FSDIBitmapDitherFS (
    FS\_DIBitmap bitmap,
    FS\_ARGB* pPalette,
    FS\_INT32 palSize,
    FS\_Rect* pRect
);
```

### Description

Floyd-Steinberg dithering for (portion of) the bitmap, using a palette.

### Parameter

---

bitmap	[In] The input bitmap.
--------	------------------------

---

pPalette	[In] The input palette color.
----------	-------------------------------

---

palSize	[In] The input palette size.
---------	------------------------------

---

pRect	[In] The input rect for dithering.
-------	------------------------------------

---

### Return

void

### Head file reference

fs\_basicTempl.h: 633

**Note:** Currently only 8-bit gray scale bitmap is supported!

## FSDIBitmapDownSampleScanline

### Syntax

```
void FSDIBitmapDownSampleScanline (
    FS\_DIBitmap bitmap,
    FS\_INT32 line,
    FS\_LPBYTE destScan,
    FS\_INT32 destBpp,
    FS\_INT32 destWidth,
    FS\_BOOL bFlipX,
    FS\_INT32 clipLeft,
    FS\_INT32 clipWidth
);
```

**Description**

Downs sample a scanline, for quick stretching. The down-sampled result would be either 8bpp (for mask and grayscale), 24bpp or 32bpp.

**Parameter**

bitmap	[In] The input bitmap.
line	[In] The zero-based line number.
destScan	[In/Out] The destination scanline buffer to receive down-sample result.
destBpp	[In] The destination bits per pixel.
destWidth	[In] The destination width of pixels in the scanline.
bFlipX	[In] Whether to flip the bitmap in x-direction.
clipLeft	[In] Clip start of the destination scanline.
clipWidth	[In] Clip width of the destination scanline.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 383

**Related method**

[FSDIBitmapGetScanline](#)

**FSDIBitmapFlipImage****Syntax**

```
FS_DIBitmap FSDIBitmapFlipImage (
    FS_DIBitmap bitmap,
    FS_BOOL bXflip,
    FS_BOOL bYflip
);
```

**Description**

Flips image. The returned bitmap must be destroyed by [FSDIBitmapDestroy](#) ().

**Parameter**


---

bitmap	[In] The input bitmap.
bXFlip	[In] Whether to flip the bitmap in x-direction.
bYFlip	[In] Whether to flip the bitmap in y-direction.

---

**Return**

A flipped bitmap.

**Head file reference**

fs\_basicTempl.h: 837

**FSDIBitmapGammaAdjust****Syntax**

```
void FSDIBitmapGammaAdjust (
    FS\_DIBitmap bitmap,
    FS\_BOOL bInvert
);
```

**Description**

Gamma adjustment.

**Parameter**


---

bitmap	[In] The input bitmap.
bInvert	[In] Whether reverse gamma adjustment or not.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 610

**FSDIBitmapGetAlphaMask****Syntax**

```
FS_DIBitmap FSDIBitmapGetAlphaMask (
    FS\_DIBitmap bitmap,
    FS\_Rect* pClip
);
```

**Description**

Gets alpha mask from a bitmap. Returns a 8bpp alpha mask. The returned value must be destroyed by [FSDIBitmapDestroy](#) ().

**Parameter**

---

bitmap	[In] The input bitmap.
--------	------------------------

---

**pClip**

[In] The clipping region of source bitmap.

---

**Return**

An alpha mask.

**Head file reference**

fs\_basicTempl.h: 736

**Note:** Applicable to Argb format only. Optionally a clipping region in bitmap coordinates can be specified to limit the size of result mask.

**FSDIBitmapGetBPP****Syntax**

```
FS_INT32 FSDIBitmapGetBPP (
    FS\_DIBitmap bitmap
);
```

**Description**

Gets the number of bits per pixel.

**Parameter**

---

bitmap	[In] The input bitmap.
--------	------------------------

---

**Return**

The number of bits per pixel.

**Head file reference**

fs\_basicTempl.h: 645

**FSDIBitmapGetBuffer****Syntax**

```
FS_LPBYTE FSDIBitmapGetBuffer (
    FS\_DIBitmap bitmap
);
```

**Description**

Gets a buffer for whole *DIB* . Only in-memory DIB can supply such a buffer.

**Parameter**

---

bitmap	[In] The input bitmap.
--------	------------------------

---

**Return**

A buffer for the whole *DIB* .

**Head file reference**

fs\_basicTempl.h: 368

**FSDIBitmapGetDPIUnit****Syntax**

```
FS_WORD FSDIBitmapGetDPIUnit (
    FS\_DIBAttribute attr
);
```

**Description**

Gets the resolution unit, described by [FS\\_DIB\\_RESUNIT](#) .

**Parameter**

---

attr	[In] The input bitmap attribute object.
------	---

---

**Return**

The resolution unit.

**Head file reference**

fs\_basicTempl.h: 923

**Related method**

[FSDIBitmapLoadInfo](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FSDIBitmapGetExifInfo****Syntax**

```
FS_BOOL FSDIBitmapGetExifInfo (
    FS\_DIBAttribute attr,
    FS\_WORD tag,
    FS\_LVOID val
);
```

**Description**

Gets the exchangeable image file information of camera in JPEG file.

**Parameter**

attr	[In] The input bitmap attribute object.
tag	[In] The input tag of the exchangeable image file information of camera in JPEG file, described by <a href="#">FSDIBEXIFTAG</a> .
val	[Out] The output value of the exchangeable image file information.

**Return**

[TRUE](#) means success, otherwise failure.

**Head file reference**

fs\_basicTempl.h: 934

**Related method**

[FSDIBitmapLoadInfo](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FSDIBitmapGetFormat****Syntax**

```
FS_DIB_Format FSDIBitmapGetFormat (
    FS_DIBitmap bitmap
);
```

**Description**

Gets the format of the bitmap.

**Parameter**

bitmap	[In] The input bitmap.
--------	------------------------

**Return**

The format of the bitmap.

**Head file reference**

fs\_basicTempl.h: 341

## FSDIBitmapGetHeight

### Syntax

```
FS_INT32 FSDIBitmapGetHeight (
    FS\_DIBitmap bitmap
);
```

### Description

Gets the height of the bitmap.

### Parameter

---

bitmap	[In] The input bitmap.
--------	------------------------

---

### Return

The height of the bitmap.

### Head file reference

fs\_basicTempl.h: 326

### Related method

[FSDIBitmapGetWidth](#)

## FSDIBitmapGetPalette

### Syntax

```
FS_ARGB* FSDIBitmapGetPalette (
    FS\_DIBitmap bitmap
);
```

### Description

Gets the palette of the bitmap.

### Parameter

---

bitmap	[In] The input bitmap.
--------	------------------------

---

### Return

The palette of the bitmap.

### Head file reference

fs\_basicTempl.h: 359

## FSDIBitmapGetPaletteArgb

**Syntax**

```
FS_ARGB FSDIBitmapGetPaletteArgb (
    FS\_DIBitmap bitmap,
    FS\_INT32 index
);
```

**Description**

Gets palette entry with specified palette entry index.

**Parameter**

---

bitmap	[In] The input bitmap.
--------	------------------------

---

index	[In] Zero-based palette entry index of the palette.
-------	---

---

**Return**

A palette entry value.

**Head file reference**

fs\_basicTempl.h: 690

**Related method**

[FSDIBitmapSetPaletteArgb](#)

**FSDIBitmapGetPaletteSize****Syntax**

```
FS_INT32 FSDIBitmapGetPaletteSize (
    FS\_DIBitmap bitmap
);
```

**Description**

Gets the number of palette entries.

**Parameter**

---

bitmap	[In] The input bitmap.
--------	------------------------

---

**Return**

The number of palette entries.

**Head file reference**

fs\_basicTempl.h: 681

**FSDIBitmapGetPitch**

**Syntax**

```
FS_DWORD FSDIBitmapGetPitch (
    FS\_DIBitmap bitmap
);
```

**Description**

Gets the specified row pitch of the bitmap.

**Parameter**

---

bitmap	[In] The input bitmap.
--------	------------------------

---

**Return**

The row pitch of the bitmap.

**Head file reference**

fs\_basicTempl.h: 350

**FSDIBitmapGetPixel****Syntax**

```
FS_DWORD FSDIBitmapGetPixel (
    FS\_DIBitmap bitmap,
    FS\_INT32 x,
    FS\_INT32 y
);
```

**Description**

Gets pixel ARGB value.

**Parameter**

---

bitmap	[In] The input bitmap.
--------	------------------------

---

---

x	[In] The x-coordinate in bitmap.
---	----------------------------------

---

---

y	[In] The y-coordinate in bitmap.
---	----------------------------------

---

**Return**

The pixel ARGB value.

**Head file reference**

fs\_basicTempl.h: 444

**Related method**

[FSDIBitmapSetPixel](#)

**Note:** For alpha mask bitmaps, only the alpha value of returned value is valid.

## FSDIBitmapGetScanline

### Syntax

```
FS_LPBYTE FSDIBitmapGetScanline (
    FS_DIBitmap bitmap,
    FS_INT32 line
);
```

### Description

Fetches a single scan line.

### Parameter

bitmap	[In] The input bitmap.
line	[In] The zero-based line number. [0, height).

### Return

A pointer to the scan line.

### Head file reference

fs\_basicTempl.h: 377

### Related method

[FSDIBitmapDownSampleScanline](#)

## FSDIBitmapGetWidth

### Syntax

```
FS_INT32 FSDIBitmapGetWidth (
    FS_DIBitmap bitmap
);
```

### Description

Gets the width of the bitmap.

### Parameter

bitmap	[In] The input bitmap.
--------	------------------------

### Return

The width of the bitmap.

**Head file reference**

fs\_basicTempl.h: 321

**Related method**[FSDIBitmapGetHeight](#)**FSDIBitmapGetXDPI****Syntax**

```
FS_INT32 FSDIBitmapGetXDPI (
    FS\_DIBAttribute attr
);
```

**Description**

Gets the horizontal resolution attribute of the bitmap.

**Parameter**

---

attr	[In] The input bitmap attribute object.
------	---

---

**Return**

The horizontal resolution, -1 means the bitmap doesn't contain this information, 0 or 1 means the user should use default DPI value.

**Head file reference**

fs\_basicTempl.h: 901

**Related method**[FSDIBitmapLoadInfo](#)

**Note:**JBIG2 cannot get DPI information now.

**Since**[SDK LATEEST VERSION > 2.0](#)**FSDIBitmapGetYDPI****Syntax**

```
FS_INT32 FSDIBitmapGetYDPI (
    FS\_DIBAttribute attr
);
```

**Description**

Gets the Vertical resolution attribute of the bitmap.

**Parameter**

---

attr	[In] The input bitmap attribute object.
------	---

---

**Return**

The vertical resolution, -1 means the bitmap doesn't contain this information, 0 or 1 means the user should use default DPI value.

**Head file reference**

fs\_basicTempl.h: 912

**Related method**

[FSDIBitmapLoadInfo](#)

**Note:**JBIG2 cannot get DPI information now.

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FSDIBitmapHasAlpha****Syntax**

```
FS_BOOL FSDIBitmapHasAlpha (
    FS_DIBitmap bitmap
);
```

**Description**

Checks if it's a bitmap with alpha channel.

**Parameter**

---

bitmap	[In] The input bitmap.
--------	------------------------

---

**Return**

Non-zero means it has a alpha channel, otherwise has not.

**Head file reference**

fs\_basicTempl.h: 663

**Note:**Alpha masks return FALSE.

**FSDIBitmapIsAlphaMask****Syntax**

```
FS_BOOL FSDIBitmapIsAlphaMask (
    FS_DIBitmap bitmap
);
```

**Description**

Checks whether the bitmap is an alpha mask (either 1bpp bitmask or 8bpp gray scale).

**Parameter**

---

bitmap	[In] The input bitmap.
--------	------------------------

---

**Return**

[TRUE](#) for bitmap being an alpha mask.

**Head file reference**

fs\_basicTempl.h: 654

**FSDIBitmapIsOpaqueImage****Syntax**

```
FS_BOOL FSDIBitmapIsOpaqueImage (
    FS\_DIBitmap bitmap
);
```

**Description**

Checks if it's a solid (opaque) image.

**Parameter**

---

bitmap	[In] The input bitmap.
--------	------------------------

---

**Return**

Non-zero means it is a solid (opaque) image, otherwise is not.

**Head file reference**

fs\_basicTempl.h: 672

**FSDIBitmapLoadChannel****Syntax**

```
void FSDIBitmapLoadChannel (
    FS\_DIBitmap bitmap,
    FS\_DIB\_Channel destChannel,
    const FS\_DIBitmap srcBitmap,
    FS\_DIB\_Channel srcChannel
);
```

**Description**

Loads a full channel from a source bitmap to this bitmap. The source bitmap can be any format, but must have specified source channel.

**Parameter**


---

bitmap	[In] The input bitmap.
destChannel	[In] The destination channel.
srcBitmap	[In] The DIB source.
srcChannel	[In] The source channel.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 469

**Related method**[FSDIBitmapLoadChannel2](#)

**Note:** If the destination channel is a color channel (R/G/B), then this bitmap must be a colored image. If the destination channel is the alpha channel, and this bitmap doesn't have an alpha channel, it will be expanded with the alpha mask loaded. If this bitmap has already an alpha channel, then the previous alpha data will be replaced. If the source bitmap doesn't have same size as this bitmap, the source bitmap will be stretched to match the destination size before channel transferring.

[FSDIBitmapLoadChannel2](#)**Syntax**

```
void FSDIBitmapLoadChannel2 (
    FS\_DIBitmap bitmap,
    FS\_DIB\_Channel destChannel,
    FS\_INT32 value
);
```

**Description**

Sets a full channel to specified value (0-255).

**Parameter**


---

bitmap	[In] The input bitmap.
destChannel	[In] The destination channel.
value	[In] The value to fill channel.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 484

**Related method**

[FSDIBitmapLoadChannel](#)

**Note:** This bitmap must be in Rgb, Rgb32, or Argb format.

**FSDIBitmapLoadFromImage****Syntax**

```
FS_DIBitmap FSDIBitmapLoadFromImage (
    FS\_FileReadHandler handler
);
```

**Description**

Loads a bitmap from a buffer.

**Parameter**

---

handler	[In] The file access handler.
---------	-------------------------------

---

**Return**

The bitmap loaded.

**Head file reference**

fs\_basicTempl.h: 958

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FSDIBitmapLoadFromPNGIcon****Syntax**

```
FS_DIBitmap FSDIBitmapLoadFromPNGIcon (
    FS\_LPCWSTR pwsFilePath
);
```

**Description**

Loads a bitmap from a PNG file.

**Parameter**

---

pwsFilePath	[In] The input PNG file path.
-------------	-------------------------------

---

**Return**

The bitmap loaded.

**Head file reference**

fs\_basicTempl.h: 848

**FSDIBitmapLoadFromPNGIcon2****Syntax**

```
FS_DIBitmap FSDIBitmapLoadFromPNGIcon2 (
    FS\_FileReadHandler handler
);
```

**Description**

Loads a bitmap from a buffer.

**Parameter**

---

handler	[In] The file access handler. The core API will take over the handler and release it. The plug-in do not need to release it.
---------	--

---

**Return**

The bitmap loaded.

**Head file reference**

fs\_basicTempl.h: 857

**FSDIBitmapLoadFromPNGIcon3****Syntax**

```
FS_DIBitmap FSDIBitmapLoadFromPNGIcon3 (
    HINSTANCE hInstance,
    FS\_LPCWSTR lpwsName
);
```

**Description**

Loads a bitmap from a instance handle.

**Parameter**

---

hInstance	[In] The input plug-in instance handle.
-----------	---

---

lpwsName	[In] The input name of PNG resource. The <a href="#">MAKEINTRESOURCE</a> macro can be used to create this value.
----------	--

---

**Return**

The bitmap loaded.

**Head file reference**

fs\_basicTempl.h: 947

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FSDIBitmapLoadInfo****Syntax**

```
FS_DIBAttribute FSDIBitmapLoadInfo (
    FS_FileReadHandler handler,
    FS_DIB_IMAGE_TYPE imageType
);
```

**Description**

Loads a bitmap attribute from a buffer.

**Parameter**

---

handler	[In] The file access handler.
---------	-------------------------------

---

imageType	[In] The image type.
-----------	----------------------

---

**Return**

The bitmap attribute loaded.

**Head file reference**

fs\_basicTempl.h: 878

**Related method**

[FSDIBitmapDestroyDIBAttribute](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FSDIBitmapMultiplyAlpha****Syntax**

```
void FSDIBitmapMultiplyAlpha (
    FS_DIBitmap bitmap,
    FS_INT32 alpha
);
```

**Description**

Multiplies alpha data with addition alpha (0-255).

**Parameter**

---

bitmap	[In] The input bitmap.
alpha	[In] The alpha value to multiply with.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 501

**Related method**

[FSDIBitmapMultiplyAlpha2](#)

**Note:** Applicable to all formats: If this is an alpha mask (1bppMask or 8bppMask), the result will be a 8bppMask with mask data modified; If this is an image without alpha channel, the bitmap will be expanded to include an alpha channel with new alpha data; If this is an image with alpha channel, then the alpha value will be multiplied into existing alpha data.

## FSDIBitmapMultiplyAlpha2

**Syntax**

```
void FSDIBitmapMultiplyAlpha2 (
    FS_DIBitmap bitmap,
    const FS_DIBitmap alphaMask
);
```

**Description**

Multiplies existing alpha data with another alpha mask.

**Parameter**

---

bitmap	[In] The input bitmap.
alphaMask	[In] The alpha mask to multiply with.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 510

**Related method**[FSDIBitmapMultiplyAlpha](#)

**Note:** Applicable to all destination formats: If this is an alpha mask (1bppMask or 8bppMask), the result will be a 8bppMask with mask data multiplied; If this is an image without alpha channel, the bitmap will be expanded to include an alpha channel with new alpha data; If this is an image with alpha channel, then the alpha value will be multiplied into existing alpha data. If the source mask doesn't have same size as this bitmap, the source mask will stretched to match the destination size before multiplying.

[FSDIBitmapNew](#)**Syntax**

```
FS_DIBitmap FSDIBitmapNew (void );
```

**Description**

Creates a bitmap. Optionally an external buffer provided by caller can be used. Using [FSDIBitmapCreate](#) () to initialize after a bitmap object is created.

**Return**

A [FS\\_DIBitmap](#) object.

**Head file reference**

fs\_basicTempl.h: 281

**Related method**[FSDIBitmapClear](#)[FSDIBitmapCreate](#)[FSDIBitmapDestroy](#)

**Note:** The buffer should be kept by caller during the existence of the bitmap.

[FSDIBitmapSetPaletteArgb](#)**Syntax**

```
void FSDIBitmapSetPaletteArgb (
    FS_DIBitmap bitmap,
    FS_INT32 index,
    FS_ARGB argb
);
```

**Description**

Changes a specified palette entry.

**Parameter**

---

bitmap	[In] The input bitmap.
--------	------------------------

---

---

index	[In] Zero-based palette entry index of the palette.
-------	---

---

argb	[In] The new value the entry.
------	-------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 696

**Related method**[FSDIBitmapGetPaletteArgb](#)**FSDIBitmapSetPixel****Syntax**

```
void FSDIBitmapSetPixel (
    FS\_DIBitmap bitmap,
    FS\_INT32 x,
    FS\_INT32 y,
    FS\_ARGB argb
);
```

**Description**

Sets pixel ARGB value.

**Parameter**


---

bitmap	[In] The input bitmap.
--------	------------------------

---

x	[In] The x-coordinate in bitmap.
---	----------------------------------

---

y	[In] The y-coordinate in bitmap.
---	----------------------------------

---

argb	[In] The input pixel ARGB value.
------	----------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 451

**Related method**[FSDIBitmapGetPixel](#)

**Note:**For alpha mask bitmaps, only the alpha value is taken.

## FSDIBitmapStretchTo

### Syntax

```
FS_DIBitmap FSDIBitmapStretchTo (
    FS_DIBitmap bitmap,
    FS_INT32 destWidth,
    FS_INT32 destHeight,
    FS_DWORD flags,
    FS_Rect* pClip
);
```

### Description

Stretches this bitmap into a new bitmap with different size. The returned bitmap must be destroyed by [FSDIBitmapDestroy \(\)](#).

### Parameter

bitmap	[In] The input bitmap.
destWidth	[In] The width of the destination bitmap.
destHeight	[In] The height of the destination bitmap.
flags	[In] Stretching flags. It can use <a href="#">FS_DIB_DOWNSAMPLE</a> and <a href="#">FS_DIB_INTERPOL</a> flags
pClip	[In] The clipping region of destination bitmap.

### Return

A new bitmap with different size.

### Head file reference

fs\_basicTempl.h: 747

**Note:**If dest width or dest height is negative, the bitmap will be flipped. If a 1bpp bitmap is stretched, it will become either a RGB bitmap, if it's a colored bitmap; or a 8bpp gray scale mask if it's a bit mask. Stretching can be done in down-sample mode, which doesn't do interpolation so significantly faster especially when stretching big images into small ones. Optionally a clipping region in result bitmap coordinates can be specified to limit the size of result bitmap.

## FSDIBitmapSwapXY

### Syntax



---

```
FS_DIBitmap FSDIBitmapSwapXY (
    FS\_DIBitmap bitmap,
    FS\_BOOL bXFlip,
    FS\_BOOL bYFlip,
    FS\_Rect* pClip
);
```

**Description**

Swaps X,Y coordinations of the bitmap. The image can also be flipped at the same time. The returned bitmap must be destroyed by [FSDIBitmapDestroy](#) ().

**Parameter**


---

bitmap	[In] The input bitmap.
bXFlip	[In] Whether to flip the bitmap in x-direction.
bYFlip	[In] Whether to flip the bitmap in y-direction.
pClip	[In] The clipping region of destination bitmap.

---

**Return**

A new bitmap.

**Head file reference**

fs\_basicTempl.h: 788

**Note:** Optionally a clipping region (in destination bitmap coordinates) can be specified to limit the size of result. This function can be used to rotate the bitmap 90 or 270 degree. Suppose the original image has the following 4 pixels: +---+ | 1 | 2 | +---+ | 3 | 4 | +---+ Then, depends on value of bXFlip and bYFlip, the result would look like this: if bXFlip = FALSE, bYFlip = FALSE: +---+ | 1 | 3 | +---+ | 2 | 4 | +---+ if bXFlip = TRUE, bYFlip = FALSE: +---+ | 3 | 1 | +---+ | 4 | 2 | +---+ if bxflip = false, byflip = true: +---+ | 2 | 4 | +---+ | 1 | 3 | +---+ if bxflip = true, byflip = true: +---+ | 4 | 2 | +---+ | 3 | 1 | +---+

**FSDIBitmapTakeOver****Syntax**

```
void FSDIBitmapTakeOver (
    FS\_DIBitmap bitmap,
    FS\_DIBitmap srcBitmap
);
```

**Description**

Takeovers a bitmap.

**Parameter**


---

bitmap	[In] The input bitmap.
--------	------------------------

---

srcBitmap	[In] The input source bitmap.
-----------	-------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 406

**Note:** After taking-over, the source bitmap will contain an empty bitmap, and this bitmap will contain all data of the source bitmap.

**FSDIBitmapTransferBitmap****Syntax**

```
void FSDIBitmapTransferBitmap (
    FS\_DIBitmap bitmap,
    FS\_INT32 destLeft,
    FS\_INT32 destTop,
    FS\_INT32 width,
    FS\_INT32 height,
    const FS\_DIBitmap srcBitmap,
    FS\_INT32 srcLeft,
    FS\_INT32 srcTop
);
```

**Description**

Transforms pixels from another bitmap into specified position.

**Parameter**


---

bitmap	[In] The input bitmap.
--------	------------------------

---

destLeft	[In] The x-coordinate in destination bitmap.
----------	--

---

destTop	[In] The y-coordinate in destination bitmap.
---------	--

---

width	[In] The area width to transfer.
-------	----------------------------------

---

height	[In] The area height to transfer.
--------	-----------------------------------

---

srcBitmap	[In] The DIB source.
-----------	----------------------

---

---

srcLeft	[In] The x-coordinate in source bitmap.
---------	---

---

srcTop	[In] The y-coordinate in source bitmap.
--------	---

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 531

**Related method**[FSDIBitmap](#)

**Note:** The destination and source bitmaps can have different format, but need to be compatible. "width" and "height" parameters can not be negative. Positions will be automatically clipped if out of source or destination bitmap region. The destination region will be replaced by data from the source bitmap.

**FSDIBitmapTransferMask****Syntax**

```
void FSDIBitmapTransferMask (
    FS\_DIBitmap bitmap,
    FS\_INT32 destLeft,
    FS\_INT32 destTop,
    FS\_INT32 width,
    FS\_INT32 height,
    const FS\_DIBitmap mask,
    FS\_ARGB argb,
    FS\_INT32 srcLeft,
    FS\_INT32 srcTop
);
```

**Description**

Transfers (portion of) an alpha mask (with source color) to this bitmap.

**Parameter**


---

bitmap	[In] The input bitmap.
--------	------------------------

---

destLeft	[In] The x-coordinate in destination bitmap.
----------	--

---

destTop	[In] The y-coordinate in destination bitmap.
---------	--

---

width	[In] The area width to transfer.
-------	----------------------------------

---

height	[In] The area height to transfer.
--------	-----------------------------------

---

---

mask	[In] The mask source.
argb	[In] The source color.
srcLeft	[In] The x-coordinate in source bitmap.
srcTop	[In] The y-coordinate in source bitmap.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 551

**Related method**[FSDIBitmapCompositeBitmap](#)[FSDIBitmapCompositeMask](#)[FSDIBitmapCompositeRect](#)

**Note:** Applicable to Argb formats only. The changed portion will be replaced (not blended). The mask parameter must point to an alpha mask bitmap (1bppMask or 8bppMask).

**FSDIBitmapTransformTo****Syntax**

```
FS_DIBitmap FSDIBitmapTransformTo (
    FS_DIBitmap bitmap,
    FS_AffineMatrix* pMatrix,
    FS_INT32* outLeft,
    FS_INT32* outTop,
    FS_DWORD flags,
    FS_Rect* pClip
);
```

**Description**

Transforms this bitmap. A new transformed bitmap is returned. The returned bitmap must be destroyed by [FSDIBitmapDestroy](#) ().

**Parameter**


---

bitmap	[In] The input bitmap.
pMatrix	[In] The transformation matrix.

---

---

outLeft	[Out] Receives x-coordinate of the left-top corner of the result bitmap in destination coords.
outTop	[Out] Receives y-coordinate of the left-top corner of the result bitmap in destination coords.
flags	[In] Stretching flags. It can use <a href="#">FS_DIB_DOWNSAMPLE</a> and <a href="#">FS_DIB_INTERPOL</a> flags.
pClip	[In] The clipping region of destination bitmap.

---

**Return**

A new transformed bitmap.

**Head file reference**

fs\_basicTempl.h: 765

**Note:** This bitmap can be colored bitmap, or an alpha mask. In case of colored bitmap, certain transformation (rotating or skewing) will cause the return bitmap as ARGB, no matter what the source bitmap format is. If a 1bpp bitmap is transformed, it will become either a RGB bitmap, if it's a colored bitmap; or a 8bpp gray scale mask if it's a bit mask. The dimension of returned bitmap always match the dimension of the matrix. Transformation can be done in down-sample mode, which doesn't do interpolation so significantly faster especially when transforming big images into small ones. Optionally a clipping region in result bitmap coordinates can be specified to limit the size of result bitmap. The position of left-top corner (in destination coordinates) of the result bitmap is also returned.

## FS\_DWordArray

### [Return from Used by](#)

### Description

A double word array. See [FSDWordArrayNew](#) , [FSDWordArrayDestroy](#) .

### Returned from

#### [FSDWordArrayNew](#)

### Used by

[FRDocConvertPdfToOtherFormat](#)  
[FPDParseGetLengthOfVersions](#)  
[FPDParseParseIndirectObjectsAtRange](#)  
[FSDWordArrayAdd](#)  
[FSDWordArrayAppend](#)  
[FSDWordArrayCopy](#)

[\*\*FSDWordArrayDestroy\*\*](#)  
[\*\*FSDWordArrayFind\*\*](#)  
[\*\*FSDWordArrayGetAt\*\*](#)  
[\*\*FSDWordArrayGetDataPtr\*\*](#)  
[\*\*FSDWordArrayGetSize\*\*](#)  
[\*\*FSDWordArrayGetUpperBound\*\*](#)  
[\*\*FSDWordArrayInsertAt\*\*](#)  
[\*\*FSDWordArrayInsertNewArrayAt\*\*](#)  
[\*\*FSDWordArrayRemoveAll\*\*](#)  
[\*\*FSDWordArrayRemoveAt\*\*](#)  
[\*\*FSDWordArraySetAt\*\*](#)  
[\*\*FSDWordArraySetAtGrow\*\*](#)  
[\*\*FSDWordArraySetSize\*\*](#)

## Functions

### Functions summary

#### [\*\*FSDWordArrayAdd\*\*](#)

Adds an element at the tail. Potentially growing the array

#### [\*\*FSDWordArrayAppend\*\*](#)

Appends an array.

#### [\*\*FSDWordArrayCopy\*\*](#)

Copies from an array.

#### [\*\*FSDWordArrayDestroy\*\*](#)

Destroys a double word array.

#### [\*\*FSDWordArrayFind\*\*](#)

Finds an element from specified position to last

#### [\*\*FSDWordArrayGetAt\*\*](#)

Retrieves an element specified by an index number.

#### [\*\*FSDWordArrayGetDataPtr\*\*](#)

Gets a pointer to the specified element in the array. Direct pointer access.

#### [\*\*FSDWordArrayGetSize\*\*](#)

Gets the number of elements in the array.

#### [\*\*FSDWordArrayGetUpperBound\*\*](#)

Gets the upper bound in the array. actually the maximum valid index.

#### [\*\*FSDWordArrayInsertAt\*\*](#)

Inserts one or more continuous element at specified position.

#### [\*\*FSDWordArrayInsertNewArrayAt\*\*](#)

Insets an array at specified position.

#### [\*\*FSDWordArrayNew\*\*](#)

Creates a new empty DWord array.

#### [\*\*FSDWordArrayRemoveAll\*\*](#)

Cleans up the array.

#### [\*\*FSDWordArrayRemoveAt\*\*](#)

Removes a number of elements at specified position.

#### [\*\*FSDWordArraySetAt\*\*](#)

Overwrites an element specified by an index number.

#### [\*\*FSDWordArraySetAtGrow\*\*](#)

Sets an element value at specified position. Potentially growing the array.

#### [\*\*FSDWordArraySetSize\*\*](#)

Changes the allocated size and the grow amount.

## Functions detail

### FSDWordArrayAdd

#### Syntax

```
FS_INT32 FSDWordArrayAdd (
    FS_DWordArray arr,
    FS_DWORD newItem
);
```

#### Description

Adds an element at the tail. Potentially growing the array

#### Parameter

---

arr	[In] The input DWord array.
-----	-----------------------------

---

newItem	[In] The input element.
---------	-------------------------

---

#### Return

The added element's index.

#### Head file reference

fs\_basicTempl.h: 1743

### FSDWordArrayAppend

#### Syntax

```
FS_INT32 FSDWordArrayAppend (
    FS_DWordArray arr,
    const FS_DWordArray srcArr
);
```

#### Description

Appends an array.

#### Parameter

---

arr	[In] The input DWord array.
-----	-----------------------------

---

srcArr	[In] The input array.
--------	-----------------------

---

#### Return

The old size in elements.

#### Head file reference

fs\_basicTempl.h: 1753

## FSDWordArrayCopy

### Syntax

```
void FSDWordArrayCopy (
    FS_DWordArray arr,
    FS_DWordArray srcArr
);
```

### Description

Copies from an array.

### Parameter

---

arr	[In] The input DWord array.
-----	-----------------------------

---

srcArr	[In] The input array.
--------	-----------------------

---

### Return

void

### Head file reference

fs\_basicTempl.h: 1763

## FSDWordArrayDestroy

### Syntax

```
void FSDWordArrayDestroy (
    FS_DWordArray arr
);
```

### Description

Destroys a double word array.

### Parameter

---

arr	[In] The input DWord array.
-----	-----------------------------

---

### Return

void

### Head file reference

fs\_basicTempl.h: 1664

## FSDWordArrayFind

**Syntax**

```
FS_INT32 FSDWordArrayFind (
    FS_DWordArray arr,
    FS_DWORD item,
    FS_INT32 nstartIndex
);
```

**Description**

Finds an element from specified position to last

**Parameter**

arr	[In] The input DWord array.
-----	-----------------------------

item	[In] The input element.
------	-------------------------

nstartIndex	[In] Specifies the zero-based index of start element to find.
-------------	---

**Return**

An index of the found element. It can be -1 for found none.

**Head file reference**

fs\_basicTempl.h: 1817

**FSDWordArrayGetAt****Syntax**

```
FS_DWORD FSDWordArrayGetAt (
    FS_DWordArray arr,
    FS_INT32 index
);
```

**Description**

Retrieves an element specified by an index number.

**Parameter**

arr	[In] The input DWord array.
-----	-----------------------------

index	[In] Specifies the zero-based index of the element.
-------	---

**Return**

An element

**Head file reference**

fs\_basicTempl.h: 1711

## FSDWordArrayGetDataPtr

### Syntax

```
FS_DWORD* FSDWordArrayGetDataPtr (
    FS_DWordArray arr,
    FS_INT32 index
);
```

### Description

Gets a pointer to the specified element in the array. Direct pointer access.

### Parameter

arr	[In] The input DWord array.
index	[In] Specifies the zero-based index of element in the array.

### Return

A pointer to the specified element.

### Head file reference

fs\_basicTempl.h: 1773

## FSDWordArrayGetSize

### Syntax

```
FS_INT32 FSDWordArrayGetSize (
    FS_DWordArray arr
);
```

### Description

Gets the number of elements in the array.

### Parameter

arr	[In] The input DWord array.
-----	-----------------------------

### Return

The number of elements in the array.

### Head file reference

fs\_basicTempl.h: 1673

## FSDWordArrayGetUpperBound

### Syntax

```
FS_INT32 FSDWordArrayGetUpperBound (
    FS_DWordArray arr
);
```

**Description**

Gets the upper bound in the array. actually the maximum valid index.

**Parameter**

---

arr	[In] The input DWord array.
-----	-----------------------------

---

**Return**

The upper bound.

**Head file reference**

fs\_basicTempl.h: 1682

**FSDWordArrayInsertAt****Syntax**

```
void FSDWordArrayInsertAt (
    FS_DWordArray arr,
    FS_INT32 index,
    FS_DWORD newItem,
    FS_INT32 nCount
);
```

**Description**

Inserts one or more continuous element at specified position.

**Parameter**

---

arr	[In] The input DWord array.
-----	-----------------------------

---

index	[In] Specifies the zero-based index in the array.
-------	---

newItem	[In] Specifies the element value to insert.
---------	---

nCount	[In] Specifies the count of the element to insert.
--------	--

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1783

**FSDWordArrayInsertNewArrayAt****Syntax**

```
void FSDWordArrayInsertNewArrayAt (
    FS_DWordArray arr,
    FS_INT32 nstartIndex,
    FS_DWordArray newArray
);
```

**Description**

Insets an array at specified position.

**Parameter**

arr	[In] The input DWord array.
nstartIndex	[In] Specifies the zero-based index of start element to insert at.
newArray	[In] The input array.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1806

**FSDWordArrayNew****Syntax**

```
FS_DWordArray FSDWordArrayNew (void );
```

**Description**

Creates a new empty DWord array.

**Return**

A new empty DWord array.

**Head file reference**

fs\_basicTempl.h: 1655

**FSDWordArrayRemoveAll****Syntax**

```
void FSDWordArrayRemoveAll (
    FS_DWordArray arr
);
```

**Description**

Cleans up the array.

**Parameter**

---

arr	[In] The input DWord array.
-----	-----------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1702

**FSDWordArrayRemoveAt****Syntax**

```
void FSDWordArrayRemoveAt (
    FS\_DWordArray arr,
    FS\_INT32 index,
    FS\_INT32 nCount
);
```

**Description**

Removes a number of elements at specified position.

**Parameter**

---

arr	[In] The input DWord array.
-----	-----------------------------

---

---

index	[In] Specifies the zero-based index in the array.
-------	---

---

---

nCount	[In] Specifies the count of element to remove.
--------	--

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1795

**FSDWordArraySetAt****Syntax**

```
void FSDWordArraySetAt (
    FS\_DWordArray arr,
```

```
FS_INT32 index,  
FS_DWORD newItem  
);
```

**Description**

Overwrites an element specified by an index number.

**Parameter**

arr	[In] The input DWord array.
index	[In] Specifies the zero-based index of the element.
newItem	[In] An element

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1721

**FSDWordArraySetAtGrow****Syntax**

```
void FSDWordArraySetAtGrow (  
    FS_DWordArray arr,  
    FS_INT32 index,  
    FS_DWORD newItem  
) ;
```

**Description**

Sets an element value at specified position. Potentially growing the array.

**Parameter**

arr	[In] The input DWord array.
index	[In] Specifies the zero-based index of element in the array.
newItem	[In] The input element.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1732

## FSDWordArraySetSize

### Syntax

```
void FSDWordArraySetSize (
    FS\_DWordArray arr,
    FS\_INT32 nNewSize,
    FS\_INT32 nGrowBy
);
```

### Description

Changes the allocated size and the grow amount.

### Parameter

arr	[In] The input DWord array.
-----	-----------------------------

nNewSize	[In] The new size in elements expected.
----------	---

nGrowBy	[In] The grow amount in elements expected, nGrowBy can be -1 for the grow amount unchanged.
---------	---

### Return

void

### Head file reference

[fs\\_basicTempl.h](#): 1691

# FS\_ExtensionHFTMgr

## Description

It is used to manage the extension HFTs. See [FSExtensionHFTMgrNewHFT](#) , [FSExtensionHFTMgrAddHFT](#) , [FSExtensionHFTMgrGetHFT](#) , [FSExtensionHFTMgrReplaceEntry](#) , [FSExtensionHFTMgrGetEntry](#) .

## Functions

### Functions summary

#### [FSExtensionHFTMgrAddHFT](#)

Adds a new function table.

#### [FSExtensionHFTMgrGetEntry](#)

Gets the entry by specified selector.

#### [FSExtensionHFTMgrGetHFT](#)

Gets the function table.

**FSExtensionHFTMgrNewHFT**

Creates a new function table which can be filled with several functions.

**FSExtensionHFTMgrReplaceEntry**

Replaces the entry by specified selector.

**Functions detail****FSExtensionHFTMgrAddHFT****Syntax**

```
FS_INT32 FSExtensionHFTMgrAddHFT (
    const FS_CHAR* hftName,
    FS_INT32 version,
    FS_HFT hft
);
```

**Description**

Adds a new function table.

**Parameter**


---

hftName	[In] The name of function table, used to identify it.
---------	---

---

version	[In] Version of this function table.
---------	--------------------------------------

---

hft	[In] The instance to be added.
-----	--------------------------------

---

**Return**

If success, return [ERR\\_ADDHFT\\_NAMEEXIST](#) (1), else(hft not exist) return [ERR\\_ADDHFT\\_UNKNOWN](#) (0).

**Head file reference**

fs\_basicTempl.h: 26

**Related method****FSExtensionHFTMgrNewHFT****FSExtensionHFTMgrGetEntry****Syntax**

```
void* FSExtensionHFTMgrGetEntry (
    FS_HFT hft,
    FS_INT32 iSel
);
```

**Description**

Gets the entry by specified selector.

**Parameter**


---

hft	[In] The function table which contains several entries.
-----	---

---

**iSel**

[In] The index of entry to be found.

**Return**

If entry exists, returns the instance, else returns [NULL](#).

**Head file reference**

fs\_basicTempl.h: 62

**Related method**

[FSExtensionHFTMgrReplaceEntry](#)

**FSExtensionHFTMgrGetHFT****Syntax**

```
FS_HFT FSExtensionHFTMgrGetHFT (
    const FS_CHAR* hftName,
    FS_INT32 version
);
```

**Description**

Gets the function table.

**Parameter**


---

hftName	[In] The name of function table, used to identify it.
---------	---

---

**version**

[In] Version of this function table.

**Return**

If exists, returns the instance, else returns [NULL](#).

**Head file reference**

fs\_basicTempl.h: 27

**Related method**

[FSExtensionHFTMgrReplaceEntry](#)

**FSExtensionHFTMgrNewHFT****Syntax**

```
FS_HFT FSExtensionHFTMgrNewHFT (
```

```
FS_INT32 numOfIntefaces  
);
```

**Description**

Creates a new function table which can be filled with several functions.

**Parameter**

---

numOfIntefaces	[In] The capacity of this new function table.
----------------	---

---

**Return**

A new function table.

**Head file reference**

fs\_basicTempl.h: 21

**Related method**

[FSExtensionHFTMgrAddHFT](#)

[FSExtensionHFTMgrGetHFT](#)

**FSExtensionHFTMgrReplaceEntry****Syntax**

```
void FSExtensionHFTMgrReplaceEntry (  
    FS_HFT hft,  
    FS_INT32 iSel,  
    void* newEntry  
);
```

**Description**

Replaces the entry by specified selector.

**Parameter**

---

hft	[In] The function table which contains the entry to be replaced.
-----	--

---

---

iSel	[In] The index of entry to be replaced.
------	---

---

---

newEntry	[In] The new entry to replaced.
----------	---------------------------------

---

**Return****Head file reference**

fs\_basicTempl.h: 50

**Related method**

[FSExtensionHFTMgrGetEntry](#)

## FS\_FileReadHandler

### [Return from Used by](#)

#### Description

File reading interface.

#### Returned from

[FPDParserGet FileAccess](#)  
[FSFileReadHandlerNew](#)  
[FSFileReadHandlerNew2](#)  
[FSFileReadHandlerNew3](#)

#### Used by

[FSDIBitmapLoadFromImage](#)  
[FSDIBitmapLoadFromPNGIcon2](#)  
[FSDIBitmapLoadInfo](#)  
[FSXMLElementparse2](#)  
[FPDDocStartProgressiveLoad](#)  
[FPDFFileSpecSetEmbeddedFile](#)  
[FPDParserStartAsynParse](#)  
[FPDParserStartParseCustomFile](#)  
[FSFileReadHandlerDestroy](#)  
[FSFileReadHandlerGetSize](#)  
[FSFileReadHandlerReadBlock](#)

## Callbacks

#### Callbacks summary

##### [FSFileRead\\_ReadBlock](#)

Read a data block from the file.

##### [FSFileRead\\_GetSize](#)

Gets total size of the file.

##### [FSFileRead\\_Release](#)

Called when to release everything.

#### Callbacks detail

##### [FSFileRead\\_ReadBlock](#)

###### Syntax

```
typedef FS_BOOL (*FSFileRead_ReadBlock)(  
    FS\_LVOID clientData,  
    void* buffer,  
    FS\_DWORD offset,
```

```
FS_DWORD size  
);
```

**Description**

Reads a data block from the file. Reads a data block from the file.

**Parameter**

clientData	[In] The user-supplied data.
buffer	[Out] Pointer to a buffer receiving read data
offset	[In] Byte offset for the block, from beginning of the file.
size	[In] Number of bytes for the block.

**Return**

TRUE for success, FALSE for failure.

**Head file reference**

fs\_basicExpT.h: 1215

**Group**

[FS\\_FileRead](#)

**FSFileRead\_GetSize****Syntax**

```
typedef FS_DWORD (*FSFileRead_GetSize)(  
    FS_LPVVOID clientData  
)
```

**Description**

Gets total size of the file. Gets total size of the file.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

**Return**

File size, in bytes. Implementation can return 0 for any error.

**Head file reference**

fs\_basicExpT.h: 1228

**Group**[FS\\_FileRead](#)**FSFileRead\_Release****Syntax**

```
typedef void (*FSFileRead_Release)(  
    FS_LPVVOID clientData  
)
```

**Description**

Called when to release everything. Called when to release everything

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void

**Head file reference**

fs\_basicExpT.h: 1241

**Group**[FS\\_FileRead](#)

## Functions

### Functions summary

**[FSFileReadHandlerDestroy](#)**

Frees file read handler.

**[FSFileReadHandlerGetSize](#)**

GetS total size of the file.

**[FSFileReadHandlerNew](#)**

Creates the file access handler.

**[FSFileReadHandlerNew2](#)**

Creates the file access handler from file path name.

**[FSFileReadHandlerNew3](#)**

Creates the file access handler from file path name(UNICODE).

**[FSFileReadHandlerReadBlock](#)**

Reads a data block from the file.

### Functions detail

**FSFileReadHandlerDestroy****Syntax**

```
void FSFileReadHandlerDestroy (
    FS\_FileReadHandler fileReadHander
);
```

**Description**

Frees file read handler.

**Parameter**

---

fileReadHander	[In] The file access handler.
----------------	-------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2577

**FSFileReadHandlerGetSize****Syntax**

```
FS_DWORD FSFileReadHandlerGetSize (
    FS\_FileReadHandler fileReadHander
);
```

**Description**

GetS total size of the file.

**Parameter**

---

fileReadHander	[In] The file access handler.
----------------	-------------------------------

---

**Return**

File size, in bytes..

**Head file reference**

fs\_basicTempl.h: 2586

**FSFileReadHandlerNew****Syntax**

```
FS_FileReadHandler FSFileReadHandlerNew (
    FS\_FileRead fileRead
);
```

**Description**

Creates the file access handler.

**Parameter**

---

fileRead	[In] The input file read callbacks.
----------	-------------------------------------

---

**Return**

The file access interface.

**Head file reference**

fs\_basicTempl.h: 2568

**FSFileReadHandlerNew2****Syntax**

```
FS_FileReadHandler FSFileReadHandlerNew2 (
    fileRead
);
```

**Description**

Creates the file access handler from file path name.

**Parameter**

---

fileRead	[In] The input file read callbacks.
----------	-------------------------------------

---

**Return**

The file access interface.

**Head file reference**

fs\_basicTempl.h: 2607

**FSFileReadHandlerNew3****Syntax**

```
FS_FileReadHandler FSFileReadHandlerNew3 (
    fileRead
);
```

**Description**

Creates the file access handler from file path name(UNICODE).

**Parameter**

---

fileRead	[In] The input file read callbacks.
----------	-------------------------------------

---

**Return**

The file access interface.

### **Head file reference**

fs\_basicTempl.h: 2616

## FSFileReadHandlerReadBlock

### **Syntax**

```
FS_BOOL FSFileReadHandlerReadBlock (
    FS\_FileReadHandler fileReadHander,
    void* buffer,
    FS\_DWORD offset,
    FS\_DWORD size
);
```

### **Description**

Reads a data block from the file.

### **Parameter**

fileReadHander	[In] The file access handler.
buffer	[In/Out] Pointer to a buffer receiving read data.
offset	[In] Byte offset for the block, from beginning of the file.
size	[In] Number of bytes for the block.

### **Return**

[TRUE](#) for success, [FALSE](#) for failure.

### **Head file reference**

fs\_basicTempl.h: 2595

## FS\_FileStream

### [Return from Used by](#)

### **Description**

The [FS\\_FileStream](#) object is used to read and write stream. See [FSFileStreamNew](#) .

### **Returned from**

### [FSFileStreamNew](#)

## [FSFileStreamNew2](#)

Used by

[FPDFFormStartGenerateContent](#)  
[FPDPageStartGenerateContent](#)  
[FSFileStreamDestroy](#)  
[FSFileStreamNew](#)

## Definitions

Definitions summary

### [FS\\_FILEMODE\\_ReadOnly](#)

The reading mode.

### [FS\\_FILEMODE\\_Truncate](#)

Clears existing content and sets the size of the file as 0. For writing mode only.

### [FS\\_FILEMODE\\_Write](#)

The writing mode. Creates automatically if the file doesn't exist.

## Definitions detail

### **FS\_FILEMODE\_ReadOnly**

#### **Syntax**

```
#define FS_FILEMODE_ReadOnly 1
```

#### **Description**

The reading mode.

#### **Group**

[FS FileMode](#)

#### **Head file reference**

fs\_basicExpT.h: 1555

### **FS\_FILEMODE\_Truncate**

#### **Syntax**

```
#define FS_FILEMODE_Truncate 2 //
```

#### **Description**

Clears existing content and sets the size of the file as 0. For writing mode only.

#### **Group**

[FS FileMode](#)

#### **Head file reference**

fs\_basicExpT.h: 1558

## FS\_FILEMODE\_Write

### Syntax

```
#define FS_FILEMODE_Write 0
```

### Description

The writing mode. Creates automatically if the file doesn't exist.

### Group

[FS FileMode](#)

### Head file reference

fs\_basicExpT.h: 1552

## Callbacks

### Callbacks summary

#### [FSFileStreamRetain](#)

Create a shared instance.

#### [FSFileStreamRelease](#)

Destroy the current instance.

#### [FSFileStreamGetSize](#)

Gets the current stream size, in bytes.

#### [FSFileStreamIsEOF](#)

Checks whether reach the end of the stream.

#### [FSFileStreamGetPosition](#)

Gets the current reading position in stream.

#### [FSFileStreamReadBlock](#)

Reads a data block from stream.

#### [FSFileStreamReadBlock2](#)

Reads a data block from stream.

#### [FSFileStreamWriteBlock](#)

Writes a block data into stream.

#### [FSFileStreamFlush](#)

Flushes internal buffer.

### Callbacks detail

#### FSFileStreamRetain

##### Syntax

```
typedef FS_FileStream (*FSFileStreamRetain)(  
    FS_LVOID clientData  
)
```

##### Description

Create a shared instance.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

The retained file stream object.

**Head file reference**

fs\_basicExpT.h: 1585

**Group**

[FS\\_FileStreamCallbacksRec](#)

**Related method**

[FSFileStreamNew](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.3](#)

FSFileStreamRelease

**Syntax**

```
typedef void (*FSFileStreamRelease)(  
    FS\_LPVVOID clientData  
)
```

**Description**

Destroy the current instance.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void.

**Head file reference**

fs\_basicExpT.h: 1598

**Group**

[FS\\_FileStreamCallbacksRec](#)

**Related method**

[FSFileStreamNew](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.3](#)

## FSFileStreamGetSize

### Syntax

```
typedef FS_INT32 (*FSFileStreamGetSize)(  
    FS\_LPVVOID clientData  
);
```

### Description

Gets the current stream size, in bytes.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

### Return

The current stream size, in bytes.

### Head file reference

fs\_basicExpT.h: 1611

### Group

[FS\\_FileStreamCallbacksRec](#)

### Related method

[FSFileStreamNew](#)

### Since

[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)

## FSFileStreamIsEOF

### Syntax

```
typedef FS_BOOL (*FSFileStreamIsEOF)(  
    FS\_LPVVOID clientData  
);
```

### Description

Checks whether reach the end of the stream.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

### Return

TRUE if reach the end of the stream, otherwise not.

### Head file reference

fs\_basicExpT.h: 1624

**Group**[FS\\_FileStreamCallbacksRec](#)**Related method**[FSFileStreamNew](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)**FSFileStreamGetPosition****Syntax**

```
typedef FS_INT32 (*FSFileStreamGetPosition)(  
    FS\_LPVVOID clientData  
);
```

**Description**

Gets the current reading position in stream.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

**Return**

The current reading position in stream.

**Head file reference**

fs\_basicExpT.h: 1637

**Group**[FS\\_FileStreamCallbacksRec](#)**Related method**[FSFileStreamNew](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)**FSFileStreamReadBlock****Syntax**

```
typedef FS_BOOL (*FSFileStreamReadBlock)(  
    FS\_LPVVOID clientData,  
    void* buffer,  
    FS\_INT32 offset,  
    unsigned int size  
);
```

**Description**

Reads a data block from stream.

**Parameter**

clientData	[In] The user-supplied data.
buffer	[In] Pointer to a buffer receiving data.
offset	[In] Byte offset from beginning of the file, the position to read data.
size	[In] Number of bytes to be read from stream.

**Return**

TRUE for success, FALSE for failure.

**Head file reference**

fs\_basicExpT.h: 1653

**Group**

[FS\\_FileStreamCallbacksRec](#)

**Related method**

[FSFileStreamNew](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)

## FSFileStreamReadBlock2

**Syntax**

```
typedef unsigned int (*FSFileStreamReadBlock2)(  
    FS\_LVOID clientData,  
    void* buffer,  
    unsigned int size  
);
```

**Description**

Reads a data block from stream.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

---

buffer	[In] Pointer to a buffer receiving data.
--------	--

---

size	[In] Number of bytes to be read from stream.
------	--

---

**Return**

The length of data stored in buffer. If returns 0, means error or no data.

**Head file reference**

fs\_basicExpT.h: 1668

**Group**

[FS\\_FileStreamCallbacksRec](#)

**Related method**

[FSFileStreamNew](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.3](#)

**FSFileStreamWriteBlock****Syntax**

```
typedef FS_BOOL (*FSFileStreamWriteBlock)(  
    FS_LVOID clientData,  
    const void* buffer,  
    FS_INT32 offset,  
    unsigned int size  
)
```

**Description**

Writes a block data into stream.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

buffer	[In] Pointer to the data block.
--------	---------------------------------

---

offset	[In] Byte offset from beginning of the file, the position to write data.
--------	--

---

size	[In] The length in bytes of the buffer.
------	---

---

**Return**

TRUE for success, FALSE for failure.

**Head file reference**

fs\_basicExpT.h: 1684

**Group**

[FS\\_FileStreamCallbacksRec](#)

**Related method**

[FSFileStreamNew](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.3](#)

## FSFileStreamFlush

**Syntax**

```
typedef FS_BOOL (*FSFileStreamFlush)(  
    FS\_LPVVOID clientData  
);
```

**Description**

Flushes internal buffer.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

TRUE for success, FALSE for failure.

**Head file reference**

fs\_basicExpT.h: 1697

**Group**

[FS\\_FileStreamCallbacksRec](#)

**Related method**

[FSFileStreamNew](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.3](#)

## Functions

### Functions summary

[\*\*FSFileStreamDestroy\*\*](#)

Destroys the file stream object.

[\*\*FSFileStreamNew\*\*](#)

Creates the file stream object.

### [FSFileStreamNew2](#)

Creates the file stream object.

## Functions detail

### FSFileStreamDestroy

#### **Syntax**

```
void FSFileStreamDestroy (
    FS\_FileStream fileStream
);
```

#### **Description**

Destroys the file stream object.

#### **Parameter**

---

fileStream	[In] The input file stream object.
------------	------------------------------------

---

#### **Return**

void

#### **Head file reference**

[fs\\_basicTempl.h](#): 3593

#### **Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)

### FSFileStreamNew

#### **Syntax**

```
FS_FileStream FSFileStreamNew (
    FS\_FileStreamCallbacks callbacks
);
```

#### **Description**

Creates the file stream object.

#### **Parameter**

---

callbacks	[In] The input callback set for file stream reading and writing.
-----------	--

---

#### **Return**

The file stream object.

#### **Head file reference**

[fs\\_basicTempl.h](#): 3571

**Since**  
[SDK LATEEST VERSION > 2.1.0.3](#)

## FSFileStreamNew2

### Syntax

```
FS_FileStream FSFileStreamNew2 (
    FS LPCWSTR filename,
    FS DWORD dwModes,
    void* reserved
);
```

### Description

Creates the file stream object.

### Parameter

filename	[In] The input file path.
dwModes	[In] The input file mode. See the definition of FS FileMode.
reserved	[In] It is a reserved parameter. Must set it as NULL.

### Return

The file stream object.

### Head file reference

fs\_basicTempl.h: 3581

### Since

[SDK LATEEST VERSION > 2.1.0.3](#)

## FS\_FileWriteHandler

### [Return from](#) [Used by](#)

### Description

file writing interface. See [FSFileWriteHandlerNew](#) , [FSFileWriteHandlerDestroy](#) .

### Returned from

### [FSFileWriteHandlerNew](#)

### Used by

---

[\*\*FSXMLElementOutputStream2\*\*](#)  
[\*\*FSFileWriteHandlerDestroy\*\*](#)

## Callbacks

### Callbacks summary

[\*\*FSFileWrite\\_GetSize\*\*](#)

It is called by foxit reader to get the total size of the file.

[\*\*FSFileWrite\\_Flush\*\*](#)

It is called by foxit reader to flush internal buffer of the file.

[\*\*FSFileWrite\\_WriteBlock\*\*](#)

It is called by foxit reader to write a block data.

[\*\*FSFileWrite\\_Release\*\*](#)

Called when to release everything.

### Callbacks detail

#### FSFileWrite\_GetSize

**Syntax**

```
typedef FS_DWORD (*FSFileWrite_GetSize)(  
    FS\_LVOID clientData  
)
```

**Description**

It is called by foxit reader to get the total size of the file.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

File size, in bytes. Implementation can return 0 for any error.

**Head file reference**

fs\_basicExpT.h: 1367

**Group**

[FS\\_FileWrite](#)

#### FSFileWrite\_Flush

**Syntax**

```
typedef FS_DWORD (*FSFileWrite_Flush)(  
    FS\_LVOID clientData  
)
```

**Description**

It is called by foxit reader to flush internal buffer of the file.

#### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

#### Return

TRUE if flushes successfully, or FALSE to indicate error.

#### Head file reference

fs\_basicExpT.h: 1378

#### Group

[FS\\_FileWrite](#)

### FSFileWrite\_WriteBlock

#### Syntax

```
typedef FS_BOOL (*FSFileWrite_WriteBlock)(
    FS\_LPVVOID clientData,
    const void* pData,
    FS\_DWORD offset,
    FS\_DWORD size
);
```

#### Description

It is called by foxit reader to write a block data.

#### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

pData	[In] The block data.
-------	----------------------

---

offset	[In] Byte offset from beginning of the file
--------	---

---

size	[In] The length in bytes of the block data.
------	---

---

#### Return

TRUE for success, or FALSE to indicate error.

#### Head file reference

fs\_basicExpT.h: 1392

#### Group

[FS\\_FileWrite](#)

## FSFileWrite\_Release

### Syntax

```
typedef void (*FSFileWrite_Release)(  
    FS_LPVVOID clientData  
)
```

### Description

Called when to release everything.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

### Return

#### Head file reference

fs\_basicExpT.h: 1405

#### Group

[FS\\_FileWrite](#)

## Functions

### Functions summary

#### [FSFileWriteHandlerDestroy](#)

Destroys the input file write handler.

#### [FSFileWriteHandlerNew](#)

Creates the file access handler for writing.

### Functions detail

#### [FSFileWriteHandlerDestroy](#)

### Syntax

```
void FSFileWriteHandlerDestroy (  
    FS_FileWriteHandler handler  
)
```

### Description

Destroys the input file write handler.

### Parameter

---

handler	[In] The input file write handler.
---------	------------------------------------

---

### Return

void

**Head file reference**

fs\_basicTempl.h: 2841

**Related method**

[FSFileWriteHandlerNew](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FSFileWriteHandlerNew**

**Syntax**

```
FS_FileWriteHandler FSFileWriteHandlerNew (
    FS\_FileWrite fileWrite
);
```

**Description**

Creates the file access handler for writing.

**Parameter**

---

fileWrite	[In] The input file write callbacks.
-----------	--------------------------------------

---

**Return**

The file access interface for writing.

**Head file reference**

fs\_basicTempl.h: 2835

**Related method**

[FSFileWriteHandlerDestroy](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

## FS\_FloatRectArray

[Return from Used by](#)

**Description**

A FS\_FloatRect array. See [FSFloatRectArrayNew](#) , [FSFloatRectArrayDestroy](#) .

**Returned from**

**[FSFloatRectArrayNew](#)**

Used by

[FPDLinkExtractGetRects](#)  
[FPDTextPageGetRectArray](#)  
[FPDTextPageGetRectsArrayByRect](#)  
[FPDTextPageFindGetRectArray](#)  
[FSFloatRectArrayAdd](#)  
[FSFloatRectArrayAppend](#)  
[FSFloatRectArrayCopy](#)  
[FSFloatRectArrayDestroy](#)  
[FSFloatRectArrayFind](#)  
[FSFloatRectArrayGetAt](#)  
[FSFloatRectArrayGetSize](#)  
[FSFloatRectArrayGetUpperBound](#)  
[FSFloatRectArrayInsertAt](#)  
[FSFloatRectArrayInsertNewArrayAt](#)  
[FSFloatRectArrayRemoveAll](#)  
[FSFloatRectArrayRemoveAt](#)  
[FSFloatRectArraySetAt](#)  
[FSFloatRectArraySetAtGrow](#)  
[FSFloatRectArraySetSize](#)

## Functions

### Functions summary

**[FSFloatRectArrayAdd](#)**

Adds an element at the tail. Potentially growing the array

**[FSFloatRectArrayAppend](#)**

Appends an array.

**[FSFloatRectArrayCopy](#)**

Copies from an array.

**[FSFloatRectArrayDestroy](#)**

Destroys the float rectangle array.

**[FSFloatRectArrayFind](#)**

Finds an element from specified position to last

**[FSFloatRectArrayGetAt](#)**

Retrieves an element specified by an index number.

**[FSFloatRectArrayGetSize](#)**

Gets the number of elements in the array.

**[FSFloatRectArrayGetUpperBound](#)**

Gets the upper bound in the array. actually the maximum valid index.

**[FSFloatRectArrayInsertAt](#)**

Insets one or more continuous element at specified position.

**[FSFloatRectArrayInsertNewArrayAt](#)**

Insets an array at specified position.

**[FSFloatRectArrayNew](#)**

Creates a new empty float rectangle array.

**[FSFloatRectArrayRemoveAll](#)**

Cleans up the array.

**FSFloatRectArrayRemoveAt**

Removes a number of elements at specified position.

**FSFloatRectArraySetAt**

Overwrites an element specified by an index number.

**FSFloatRectArraySetAtGrow**

Sets an element value at specified position. Potentially growing the array.

**FSFloatRectArraySetSize**

Changes the allocated size and the grow amount.

**Functions detail****FSFloatRectArrayAdd****Syntax**

```
FS_INT32 FSFloatRectArrayAdd (
    FS_FloatRectArray arr,
    FS_FloatRect newItem
);
```

**Description**

Adds an element at the tail. Potentially growing the array

**Parameter**

arr	[In] The input float rectangle array.
-----	---------------------------------------

newItem	[In] The input element.
---------	-------------------------

**Return**

The added element's index.

**Head file reference**

fs\_basicTempl.h: 2278

**FSFloatRectArrayAppend****Syntax**

```
FS_INT32 FSFloatRectArrayAppend (
    FS_FloatRectArray arr,
    const FS_FloatRectArray srcArr
);
```

**Description**

Appends an array.

**Parameter**

---

arr	[In] The input float rectangle array.
-----	---------------------------------------

---

srcArr	[In] The input array.
--------	-----------------------

---

**Return**

The old size in elements.

**Head file reference**

fs\_basicTempl.h: 2288

**FSFloatRectArrayCopy****Syntax**

```
void FSFloatRectArrayCopy (
    FS\_FloatRectArray arr,
    FS\_FloatRectArray srcArr
);
```

**Description**

Copies from an array.

**Parameter**


---

arr	[In] The input float rectangle array.
-----	---------------------------------------

---

srcArr	[In] The input array.
--------	-----------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2298

**FSFloatRectArrayDestroy****Syntax**

```
void FSFloatRectArrayDestroy (
    FS\_FloatRectArray arr
);
```

**Description**

Destroys the float rectangle array.

**Parameter**

---

arr	[In] The input float rectangle array.
-----	---------------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2199

**FSFloatRectArrayFind****Syntax**

```
FS_INT32 FSFloatRectArrayFind (
    FS_FloatRectArray arr,
    FS_FloatRect item,
    FS_INT32 nstartIndex
);
```

**Description**

Finds an element from specified position to last

**Parameter**


---

arr	[In] The input float rectangle array.
-----	---------------------------------------

---

item	[In] The input element.
------	-------------------------

---

nstartIndex	[In] Specifies the zero-based index of start element to find.
-------------	---

---

**Return**

An index of the found element. It can be -1 for found none.

**Head file reference**

fs\_basicTempl.h: 2342

**FSFloatRectArrayGetAt****Syntax**

```
FS_FloatRect FSFloatRectArrayGetAt (
    FS_FloatRectArray arr,
    FS_INT32 index
);
```

**Description**

Retrieves an element specified by an index number.

**Parameter**

---

arr	[In] The input float rectangle array.
-----	---------------------------------------

---

index	[In] Specifies the zero-based index of the element.
-------	---

---

**Return**

An element

**Head file reference**

fs\_basicTempl.h: 2246

**FSFloatRectArrayGetSize****Syntax**

```
FS_INT32 FSFloatRectArrayGetSize (
    FS\_FloatRectArray arr
);
```

**Description**

Gets the number of elements in the array.

**Parameter**

---

arr	[In] The input float rectangle array.
-----	---------------------------------------

---

**Return**

The number of elements in the array.

**Head file reference**

fs\_basicTempl.h: 2208

**FSFloatRectArrayGetUpperBound****Syntax**

```
FS_INT32 FSFloatRectArrayGetUpperBound (
    FS\_FloatRectArray arr
);
```

**Description**

Gets the upper bound in the array. actually the maximum valid index.

**Parameter**

---

arr	[In] The input float rectangle array.
-----	---------------------------------------

---

**Return**

The upper bound.

**Head file reference**

fs\_basicTempl.h: 2217

**FSFloatRectArrayInsertAt****Syntax**

```
void FSFloatRectArrayInsertAt (
    FS\_FloatRectArray arr,
    FS\_INT32 index,
    FS\_FloatRect newItem,
    FS\_INT32 nCount
);
```

**Description**

Insets one or more continuous element at specified position.

**Parameter**

---

arr	[In] The input float rectangle array.
-----	---------------------------------------

---

index	[In] Specifies the zero-based index in the array.
-------	---

---

newItem	[In] Specifies the element value to insert.
---------	---

---

nCount	[In] Specifies the count of the element to insert.
--------	--

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2308

**FSFloatRectArrayInsertNewArrayAt****Syntax**

```
void FSFloatRectArrayInsertNewArrayAt (
    FS\_FloatRectArray arr,
    FS\_INT32 nstartIndex,
    FS\_FloatRectArray newArray
);
```

**Description**

Insets an array at specified position.

**Parameter**

---

arr	[In] The input float rectangle array.
nstartIndex	[In] Specifies the zero-based index of start element to insert at.
newArray	[In] The input array.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2331

**FSFloatRectArrayNew****Syntax**

```
FS_FloatRectArray FSFloatRectArrayNew (void );
```

**Description**

Creates a new empty float rectangle array.

**Return**

A new empty float rectangle array.

**Head file reference**

fs\_basicTempl.h: 2190

**FSFloatRectArrayRemoveAll****Syntax**

```
void FSFloatRectArrayRemoveAll (
    FS\_FloatRectArray arr
);
```

**Description**

Cleans up the array.

**Parameter**


---

arr	[In] The input float rectangle array.
-----	---------------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2237

## FSFloatRectArrayRemoveAt

### Syntax

```
void FSFloatRectArrayRemoveAt (
    FS\_FloatRectArray arr,
    FS\_INT32 index,
    FS\_INT32 nCount
);
```

### Description

Removes a number of elements at specified position.

### Parameter

---

arr	[In] The input float rectangle array.
-----	---------------------------------------

---

index	[In] Specifies the zero-based index in the array.
-------	---

---

nCount	[In] Specifies the count of element to remove.
--------	--

---

### Return

void

### Head file reference

fs\_basicTempl.h: 2320

## FSFloatRectArraySetAt

### Syntax

```
void FSFloatRectArraySetAt (
    FS\_FloatRectArray arr,
    FS\_INT32 index,
    FS\_FloatRect newItem
);
```

### Description

Overwrites an element specified by an index number.

### Parameter

---

arr	[In] The input float rectangle array.
-----	---------------------------------------

---

index	[In] Specifies the zero-based index of the element.
-------	---

---

newItem	[In] An element
---------	-----------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2256

**FSFloatRectArraySetAtGrow****Syntax**

```
void FSFloatRectArraySetAtGrow (
    FS\_FloatRectArray arr,
    FS\_INT32 index,
    FS\_FloatRect newItem
);
```

**Description**

Sets an element value at specified position. Potentially growing the array.

**Parameter**

---

arr	[In] The input float rectangle array.
-----	---------------------------------------

---

index	[In] Specifies the zero-based index of element in the array.
-------	--

---

newItem	[In] The input element.
---------	-------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2267

**FSFloatRectArraySetSize****Syntax**

```
void FSFloatRectArraySetSize (
    FS\_FloatRectArray arr,
    FS\_INT32 nNewSize,
    FS\_INT32 nGrowBy
);
```

**Description**

Changes the allocated size and the grow amount.

**Parameter**


---

arr	[In] The input float rectangle array.
nNewSize	[In] The new size in elements expected.
nGrowBy	[In] The grow amount in elements expected, nGrowBy can be -1 for the grow amount unchanged.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2226

# FS\_GUID

## Description

## Functions

### Functions summary

**[FSGUIDCreate](#)**

Creates a GUID, version 4.

**[FSGUIDToString](#)**

FormatS GUID to a string.

### Functions detail

**FSGUIDCreate****Syntax**

```
void FSGUIDCreate (
    FS\_LPGUID pGUID
);
```

**Description**

Creates a GUID, version 4.

**Parameter**


---

pGUID	[Out] Pointer to store the returned GUID value, cannot be NULL.
-------	---

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3612

**Since**

[SDK LATEEST VERSION > 7.1](#)

**FSGUIDToString****Syntax**

```
void FSGUIDToString (
    FS\_LPGUID pGUID,
    FS\_ByteString* outStr,
    FS\_BOOL bSeparator
);
```

**Description**

Formats GUID to a string.

**Parameter**

---

pGUID	[In] Pointer to a valid GUID value, cannot be NULL.
-------	---

---

outStr	[Out] It receives the GUID string.
--------	------------------------------------

---

bSeparator	[In] If TRUE, GUID string will include '-' separating characters.
------------	---

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3622

**Since**

[SDK LATEEST VERSION > 7.1](#)

**FS\_HFT**

[Return from Used by](#)

**Description**

An object that describes a set of exported functions. It is an array of function pointers. See [FSExtensionHFTMgrNewHFT](#) , [FSExtensionHFTMgrGetHFT](#) , [FSExtensionHFTMgrReplaceEntry](#) .

#### Returned from

[FSExtensionHFTMgrGetHFT](#)  
[FSExtensionHFTMgrNewHFT](#)

#### Used by

[FSExtensionHFTMgrAddHFT](#)  
[FSExtensionHFTMgrGetEntry](#)  
[FSExtensionHFTMgrReplaceEntry](#)

## FS\_MapByteStringToPtr

[Return from](#) [Used by](#)

#### Description

FS\_ByteString TO POINTER MAP. See [FSMapByteStringToPtrNew](#) , [FSMapByteStringToPtrDestroy](#) .

#### Returned from

[FSMapByteStringToPtrNew](#)

#### Used by

[FSMapByteStringToPtrDestroy](#)  
[FSMapByteStringToPtrGetCount](#)  
[FSMapByteStringToPtrGetHashTableSize](#)  
[FSMapByteStringToPtrGetNextAssoc](#)  
[FSMapByteStringToPtrGetNextValue](#)  
[FSMapByteStringToPtrGetStartPosition](#)  
[FSMapByteStringToPtrHashKey](#)  
[FSMapByteStringToPtrInitHashTable](#)  
[FSMapByteStringToPtrIsEmpty](#)  
[FSMapByteStringToPtrLookup](#)  
[FSMapByteStringToPtrRemoveAll](#)  
[FSMapByteStringToPtrRemoveKey](#)  
[FSMapByteStringToPtrSetAt](#)

## Functions

### Functions summary

[FSMapByteStringToPtrDestroy](#)

Destroys the bytestring-to-ptr map.

[FSMapByteStringToPtrGetCount](#)

Gets the number of elements.

**[FSMapByteStringToPtrGetHashTableSize](#)**

Gets the internal hash table size. Advanced features for derived classes.

**[FSMapByteStringToPtrGetNextAssoc](#)**

Gets the current association and sets the position to next association.

**[FSMapByteStringToPtrGetNextValue](#)**

Gets the current value and sets the position to next association.

**[FSMapByteStringToPtrGetStartPosition](#)**

Gets the first key-value pair position, iterating all (key, value) pairs.

**[FSMapByteStringToPtrHashKey](#)**

Routine used to user-provided hash keys.

**[FSMapByteStringToPtrInitHashTable](#)**

Initializes the hash table.

**[FSMapByteStringToPtrIsEmpty](#)**

Tests whether the *map* is empty.

**[FSMapByteStringToPtrLookup](#)**

Looks up by a key.

**[FSMapByteStringToPtrNew](#)**

Creates an empty bytestring-to-ptr map with specified block size.

**[FSMapByteStringToPtrRemoveAll](#)**

Removes all the (key, value) pairs in the map.

**[FSMapByteStringToPtrRemoveKey](#)**

Removes existing (key, value) pair.

**[FSMapByteStringToPtrSetAt](#)**

Adds a new (key, value) pair. Adds if not exist, otherwise modify.

## Functions detail

### **[FSMapByteStringToPtrDestroy](#)**

#### **Syntax**

```
void FSMapByteStringToPtrDestroy (
    FS\_MapByteStringToPtr map
);
```

#### **Description**

Destroys the bytestring-to-ptr map.

#### **Parameter**

---

map	[In] The input bytestring-to-ptr map.
-----	---------------------------------------

---

#### **Return**

void

#### **Head file reference**

fs\_basicTempl.h: 3728

#### **Since**

[SDK LATEEST VERSION > 7.3.1](#)

## FSMapByteStringToPtrGetCount

### Syntax

```
FS_INT32 FSMapByteStringToPtrGetCount (
    FS\_MapByteStringToPtr map
);
```

### Description

Gets the number of elements.

### Parameter

---

map	[In] The input bytestring-to-ptr map.
-----	---------------------------------------

---

### Return

The number of elements in the map.

### Head file reference

fs\_basicTempl.h: 3738

### Since

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

## FSMapByteStringToPtrGetHashTableSize

### Syntax

```
FS_DWORD FSMapByteStringToPtrGetHashTableSize (
    FS\_MapByteStringToPtr map
);
```

### Description

Gets the internal hash table size. Advanced features for derived classes.

### Parameter

---

map	[In] The input bytestring-to-ptr map.
-----	---------------------------------------

---

### Return

The hash table size.

### Head file reference

fs\_basicTempl.h: 3837

### Since

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

## FSMapByteStringToPtrGetNextAssoc

**Syntax**

```
void FSMapByteStringToPtrGetNextAssoc (
    FS\_MapByteStringToPtr map,
    FS\_POSITION* inOutNextPosition,
    FS\_ByteString* outKey,
    void** outValue
);
```

**Description**

Gets the current association and sets the position to next association.

**Parameter**

map	[In] The input bytestring-to-ptr map.
inOutNextPosition	[In/Out] Input a position, and receive the next association position.
outKey	[Out] (Filled by this method)Receive a key.
outValue	[Out] (Filled by this method)Receive a value.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3813

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FSMapByteStringToPtrGetNextValue****Syntax**

```
FS_LPVOID FSMapByteStringToPtrGetNextValue (
    FS\_MapByteStringToPtr map,
    FS\_POSITION* inOutNextPosition
);
```

**Description**

Gets the current value and sets the position to next association.

**Parameter**

map	[In] The input bytestring-to-ptr map.
-----	---------------------------------------

---

inOutNextPosition	[In/Out] Input a position, and receive the next association position.
-------------------	---

---

**Return**

The value.

**Head file reference**

fs\_basicTempl.h: 3826

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FSMapByteStringToPtrGetStartPosition****Syntax**

```
FS_POSITION FSMapByteStringToPtrGetStartPosition (
    FS\_MapByteStringToPtr map
);
```

**Description**

Gets the first key-value pair position, iterating all (key, value) pairs.

**Parameter**


---

map	[In] The input bytestring-to-ptr map.
-----	---------------------------------------

---

**Return**

The first key-value pair position in the map

**Head file reference**

fs\_basicTempl.h: 3803

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FSMapByteStringToPtrHashKey****Syntax**

```
FS_DWORD FSMapByteStringToPtrHashKey (
    FS\_MapByteStringToPtr map,
    FS\_ByteString key
);
```

**Description**

Routine used to user-provided hash keys.

**Parameter**

---

map	[In] The input bytestring-to-ptr map.
-----	---------------------------------------

---

key	[In] The key used to produce the hash key.
-----	--

---

**Return**

A hash value.

**Head file reference**

fs\_basicTempl.h: 3859

**Note:** Overwrite-able: special non-virtual (see map implementation for details).

**Since**

[SDK LATEEST VERSION > 7.3.1](#)

**FSMapByteStringToPtrInitHashTable****Syntax**

```
void FSMapByteStringToPtrInitHashTable (
    FS\_MapByteStringToPtr map,
    FS\_DWORD hashSize,
    FS\_BOOL bAllocNow
);
```

**Description**

Initializes the hash table.

**Parameter**


---

map	[In] The input bytestring-to-ptr map.
-----	---------------------------------------

---

hashSize	[In] Initializes the hash table size.
----------	---------------------------------------

---

bAllocNow	[In] Does it Now allocate the hash table? No-zero means yes, otherwise not. Sets it TRUE as default.
-----------	--

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3847

**Since**

[SDK LATEEST VERSION > 7.3.1](#)

**FSMapByteStringToPtrIsEmpty**

**Syntax**

```
FS_BOOL FSMapByteStringToPtrIsEmpty (
    FS\_MapByteStringToPtr map
);
```

**Description**

Tests whether the *map* is empty.

**Parameter**

---

map	[In] The input bytestring-to-ptr map.
-----	---------------------------------------

---

**Return**

[TRUE](#) for map being empty. [FALSE](#) otherwise.

**Head file reference**

fs\_basicTempl.h: 3748

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FSMapByteStringToPtrLookup****Syntax**

```
FS_BOOL FSMapByteStringToPtrLookup (
    FS\_MapByteStringToPtr map,
    FS\_ByteString key,
    void** outValue
);
```

**Description**

Looks up by a key.

**Parameter**

---

map	[In] The input bytestring-to-ptr map.
-----	---------------------------------------

---

---

key	[In] The key to lookup.
-----	-------------------------

---

---

outValue	[Out] A ref of a typeless pointer to receive the found value.
----------	---

---

**Return**

[TRUE](#) for value being found. [FALSE](#) otherwise.

**Head file reference**

fs\_basicTempl.h: 3758

**Since**[SDK LATEEST VERSION > 7.3.1](#)

## FSMapByteStringToPtrNew

**Syntax**

```
FS_MapByteStringToPtr FSMapByteStringToPtrNew (
    FS_INT32 nBlockSize
);
```

**Description**

Creates an empty bytestring-to-ptr map with specified block size.

**Parameter**

---

nBlockSize	[In] The internal block. Sets it 10 as default.
------------	---

---

**Return**

An empty bytestring-to-ptr map.

**Head file reference**

fs\_basicTempl.h: 3718

**Since**[SDK LATEEST VERSION > 7.3.1](#)

## FSMapByteStringToPtrRemoveAll

**Syntax**

```
void FSMapByteStringToPtrRemoveAll (
    FS_MapByteStringToPtr map
);
```

**Description**

Removes all the (key, value) pairs in the map.

**Parameter**

---

map	[In] The input bytestring-to-ptr map.
-----	---------------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3793

**Since**[SDK LATEEST VERSION > 7.3.1](#)

## FSMapByteStringToPtrRemoveKey

**Syntax**

```
FS_BOOL FSMapByteStringToPtrRemoveKey (
    FS\_MapByteStringToPtr map,
    FS\_ByteString key
);
```

**Description**

Removes existing (key, value) pair.

**Parameter**

---

map	[In] The input bytestring-to-ptr map.
-----	---------------------------------------

---

key	[In] The key to remove.
-----	-------------------------

---

**Return**

[TRUE](#) for success, otherwise [FALSE](#).

**Head file reference**

fs\_basicTempl.h: 3782

**Since**[SDK LATEEST VERSION > 7.3.1](#)

## FSMapByteStringToPtrSetAt

**Syntax**

```
void FSMapByteStringToPtrSetAt (
    FS\_MapByteStringToPtr map,
    FS\_ByteString key,
    void* newValue
);
```

**Description**

Adds a new (key, value) pair. Adds if not exist, otherwise modify.

**Parameter**

---

map	[In] The input bytestring-to-ptr map.
-----	---------------------------------------

---

---

key	[In] The key to specify a position.
newValue	[In] The new value.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3770

**Since**[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

## FS\_MapPtrToPtr

**[Return from Used by](#)****Description**

POINTER/DWORD TO POINTER/DWORD MAP. See [FSMapPtrToPtrNew](#) , [FSMapPtrToPtrDestroy](#) .

**Returned from****[FSMapPtrToPtrNew](#)****Used by**

[FRAppCollectDocActionData](#)  
[FPDFDFDocImportExternalObject](#)  
[FPDDocImportExternalObject](#)  
[FPDFFormRealizeResource](#)  
[FPDPageRealizeResource](#)  
[FSMapPtrToPtrDestroy](#)  
[FSMapPtrToPtrGetCount](#)  
[FSMapPtrToPtrGetHashTableSize](#)  
[FSMapPtrToPtrGetNextAssoc](#)  
[FSMapPtrToPtrGetStartPosition](#)  
[FSMapPtrToPtrGetValueAt](#)  
[FSMapPtrToPtrInitHashTable](#)  
[FSMapPtrToPtrIsEmpty](#)  
[FSMapPtrToPtrLookup](#)  
[FSMapPtrToPtrRemoveAll](#)  
[FSMapPtrToPtrRemoveKey](#)  
[FSMapPtrToPtrSetAt](#)

## Functions

**Functions summary**

**FSMapPtrToPtrDestroy**

Destroys a ptr-to-ptr map.

**FSMapPtrToPtrGetCount**

Gets the number of elements.

**FSMapPtrToPtrGetHashTableSize**

Gets the internal hash table size. Advanced features for derived classes.

**FSMapPtrToPtrGetNextAssoc**

Gets the current association and sets the position to next association.

**FSMapPtrToPtrGetStartPosition**

Gets the first key-value pair position, iterating all (key, value) pairs.

**FSMapPtrToPtrGetValueAt**

Retrieves a value pointer by a key

**FSMapPtrToPtrInitHashTable**

Initializes the hash table

**FSMapPtrToPtrIsEmpty**

Tests whether the *map* is empty.

**FSMapPtrToPtrLookup**

Lookups by a key.

**FSMapPtrToPtrNew**

Creates an empty ptr-to-ptr map.

**FSMapPtrToPtrRemoveAll**

Removes all the (key, value) pairs in the map.

**FSMapPtrToPtrRemoveKey**

Removes existing (key, value) pair.

**FSMapPtrToPtrSetAt**

Adds a new (key, value) pair. Adds if not exist, otherwise modify.

## Functions detail

### FSMapPtrToPtrDestroy

#### Syntax

```
void FSMapPtrToPtrDestroy (
    FS\_MapPtrToPtr map
);
```

#### Description

Destroys a ptr-to-ptr map.

#### Parameter

---

map	[In] The input ptr-to-ptr map.
-----	--------------------------------

---

#### Return

void

#### Head file reference

fs\_basicTempl.h: 985

## FSMapPtrToPtrGetCount

### Syntax

```
FS_INT32 FSMapPtrToPtrGetCount (
    FS\_MapPtrToPtr map
);
```

### Description

Gets the number of elements.

### Parameter

---

map	[In] The input ptr-to-ptr map.
-----	--------------------------------

---

### Return

The number of elements in the map.

### Head file reference

fs\_basicTempl.h: 994

## FSMapPtrToPtrGetHashTableSize

### Syntax

```
FS_DWORD FSMapPtrToPtrGetHashTableSize (
    FS\_MapPtrToPtr map
);
```

### Description

Gets the internal hash table size. Advanced features for derived classes.

### Parameter

---

map	[In] The input ptr-to-ptr map.
-----	--------------------------------

---

### Return

The hash table size.

### Head file reference

fs\_basicTempl.h: 1084

## FSMapPtrToPtrGetNextAssoc

### Syntax

```
void FSMapPtrToPtrGetNextAssoc (
    FS\_MapPtrToPtr map,
    FS\_POSITION* inOutNextPosition,
    void** outKey,
    void** outValue
```

);

**Description**

Gets the current association and sets the position to next association.

**Parameter**

map	[In] The input ptr-to-ptr map.
inOutNextPosition	[In/Out] Input a position, and receive the next association position.
outKey	[Out] (Filled by this method)Receive a key.
outValue	[Out] (Filled by this method)Receive a value.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1072

**FSMapPtrToPtrGetStartPosition****Syntax**

```
FS_POSITION FSMapPtrToPtrGetStartPosition (
    FS\_MapPtrToPtr map
);
```

**Description**

Gets the first key-value pair position, iterating all (key, value) pairs.

**Parameter**

map	[In] The input ptr-to-ptr map.
-----	--------------------------------

**Return**

The first key-value pair position in the map

**Head file reference**

fs\_basicTempl.h: 1063

**FSMapPtrToPtrGetValueAt**

**Syntax**

```
void* FSMapPtrToPtrGetValueAt (
    FS\_MapPtrToPtr map,
    void* key
);
```

**Description**

Retrieves a value pointer by a key

**Parameter**

map	[In] The input ptr-to-ptr map.
-----	--------------------------------

key	[In] The key to specify a value.
-----	----------------------------------

**Return**

A value. [NULL](#) if the key is not found.

**Head file reference**

`fs_basicTempl.h: 1023`

**FSMapPtrToPtrInitHashTable****Syntax**

```
void FSMapPtrToPtrInitHashTable (
    FS\_MapPtrToPtr map,
    FS\_DWORD hashSize,
    FS\_BOOL bAllocNow
);
```

**Description**

Initializes the hash table

**Parameter**

map	[In] The input ptr-to-ptr map.
-----	--------------------------------

hashSize	[In] Initializes the hash table size.
----------	---------------------------------------

bAllocNow	[In] Does it Now allocate the hash table? No-zero means yes, otherwise not.
-----------	---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1093

## FSMapPtrToPtrIsEmpty

### Syntax

```
FS_BOOL FSMapPtrToPtrIsEmpty (
    FS\_MapPtrToPtr map
);
```

### Description

Tests whether the *map* is empty.

### Parameter

---

map	[In] The input ptr-to-ptr map.
-----	--------------------------------

---

### Return

[TRUE](#) for map being empty. [FALSE](#) otherwise.

### Head file reference

fs\_basicTempl.h: 1003

## FSMapPtrToPtrLookup

### Syntax

```
FS_BOOL FSMapPtrToPtrLookup (
    FS\_MapPtrToPtr map,
    void* key,
    void** outValue
);
```

### Description

Lookups by a key.

### Parameter

---

map	[In] The input ptr-to-ptr map.
-----	--------------------------------

---

key	[In] The key to lookup.
-----	-------------------------

---

outValue	[Out] A ref of a typeless pointer to receive the found value.
----------	---

---

### Return

[TRUE](#) for value being found. [FALSE](#) otherwise.

**Head file reference**

fs\_basicTempl.h: 1012

**FSMapPtrToPtrNew****Syntax**

```
FS_MapPtrToPtr FSMapPtrToPtrNew (
    FS_INT32 nBlockSize
);
```

**Description**

Creates an empty ptr-to-ptr map.

**Parameter**

---

nBlockSize	[In] The internal block
------------	-------------------------

---

**Return**

An empty ptr-to-ptr map.

**Head file reference**

fs\_basicTempl.h: 976

**FSMapPtrToPtrRemoveAll****Syntax**

```
void FSMapPtrToPtrRemoveAll (
    FS_MapPtrToPtr map
);
```

**Description**

Removes all the (key, value) pairs in the map.

**Parameter**

---

map	[In] The input ptr-to-ptr map.
-----	--------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1054

**FSMapPtrToPtrRemoveKey****Syntax**

```
FS_BOOL FSMapPtrToPtrRemoveKey (
    FS\_MapPtrToPtr map,
    void* key
);
```

**Description**

Removes existing (key, value) pair.

**Parameter**

---

map	[In] The input ptr-to-ptr map.
-----	--------------------------------

---

key	[In] The key to remove.
-----	-------------------------

---

**Return**

[TRUE](#) for success, otherwise [FALSE](#).

**Head file reference**

fs\_basicTempl.h: 1044

**FSMapPtrToPtrSetAt****Syntax**

```
void FSMapPtrToPtrSetAt (
    FS\_MapPtrToPtr map,
    void* key,
    void* newValue
);
```

**Description**

Adds a new (key, value) pair. Adds if not exist, otherwise modify.

**Parameter**

---

map	[In] The input ptr-to-ptr map.
-----	--------------------------------

---

key	[In] The key to specify a position.
-----	-------------------------------------

---

newValue	[In] The new value.
----------	---------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1033

## FS\_PauseHandler

### [Return from Used by](#)

#### Description

An object preparing a simple pause instance.

#### Returned from

### [FSPauseHandlerCreate](#)

#### Used by

[FPDDocContinueLoad](#)  
[FPDDocStartProgressiveLoad](#)  
[FPDFFormContinueGenerateContent](#)  
[FPDFFormContinueParse](#)  
[FPDPageContinueGenerateContent](#)  
[FPDPageContinueParse](#)  
[FPDProgressiveRenderContinue](#)  
[FPDProgressiveRenderStart](#)  
[FPDProgressiveSearchFindFrom](#)  
[FPDUnencryptedWrapperCreatorContinue](#)  
[FPDWrapperDocContinue](#)  
[FPDWrapperDocStartGetPayload](#)  
[FSPauseHandlerDestroy](#)

## Callbacks

### Callbacks summary

#### [NeedToPauseNow](#)

Prototype of callback function.

### Callbacks detail

#### NeedToPauseNow

##### Syntax

```
typedef FS_BOOL (*NeedToPauseNow)(  
    void  
)
```

##### Description

Prototype of callback function.

##### Parameter

---

void	[In]
------	------

---

**Return**

Non-zero means we need, otherwise we need not.

**Head file reference**

fs\_basicExpT.h: 1178

**Group**

[FS\\_Pause](#)

## Functions

### Functions summary

[\*\*FSPauseHandlerCreate\*\*](#)

Creates the pause handler.

[\*\*FSPauseHandlerDestroy\*\*](#)

Destroys the pause handler.

### Functions detail

**FSPauseHandlerCreate**

**Syntax**

```
FS_PauseHandler FSPauseHandlerCreate (
    FS\_Pause pause
);
```

**Description**

Creates the pause handler.

**Parameter**

---

pause	[In] The input pause handler structure.
-------	---

---

**Return**

The newly created pause handler.

**Head file reference**

fs\_basicTempl.h: 2542

**FSPauseHandlerDestroy**

**Syntax**

```
void FSPauseHandlerDestroy (
    FS\_PauseHandler pauseHandler
);
```

**Description**

Destroys the pause handler.

**Parameter**

---

pauseHandler	[In] The input pause handler to be destroyed.
--------------	---

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2551

## FS\_PtrArray

**Return from Used by****Description**

A typeless pointer array. See [FSPtrArrayNew](#) , [FSPtrArrayDestroy](#) .

**Returned from****FSPtrArrayNew****Used by**

[FRDocLoadAnnots](#)  
[FRDocViewShowAnnots](#)  
[FRPageViewDidContentChanged3](#)  
[FPDFDFDocExportAnnotToPDFPage](#)  
[FPDACTIONGetOCGStates](#)  
[FPDACTIONFieldsGetAllFields](#)  
[FPDDocBuildResourceList](#)  
[FPDInterFormExportToFDF2](#)  
[FPDOCPropertiesGetOCGroups](#)  
[FPDOCPropertiesRetrieveOCGPages](#)  
[FSPtrArrayAdd](#)  
[FSPtrArrayAppend](#)  
[FSPtrArrayCopy](#)  
[FSPtrArrayDestroy](#)  
[FSPtrArrayFind](#)  
[FSPtrArrayGetAt](#)  
[FSPtrArrayGetDataPtr](#)  
[FSPtrArrayGetSize](#)  
[FSPtrArrayGetUpperBound](#)  
[FSPtrArrayInsertAt](#)  
[FSPtrArrayInsertNewArrayAt](#)  
[FSPtrArrayRemoveAll](#)

[FSPtrArrayRemoveAt](#)  
[FSPtrArraySetAt](#)  
[FSPtrArraySetAtGrow](#)  
[FSPtrArraySetSize](#)

## Functions

### Functions summary

#### [FSPtrArrayAdd](#)

Adds an element at the tail. Potentially grow the array.

#### [FSPtrArrayAppend](#)

Appends an array.

#### [FSPtrArrayCopy](#)

Copies from an array.

#### [FSPtrArrayDestroy](#)

Destroys a pointer array.

#### [FSPtrArrayFind](#)

Finds an element from specified position to last

#### [FSPtrArrayGetAt](#)

Retrieves an element specified by an index number.

#### [FSPtrArrayGetDataPtr](#)

Gets a pointer to the specified element in the array. Direct pointer access.

#### [FSPtrArrayGetSize](#)

Gets the number of elements in the array.

#### [FSPtrArrayGetUpperBound](#)

Gets the upper bound in the array, actually the maximum valid index.

#### [FSPtrArrayInsertAt](#)

Inserts one or more continuous element at specified position.

#### [FSPtrArrayInsertNewArrayAt](#)

Inserts an array at specified position.

#### [FSPtrArrayNew](#)

Creates a new empty pointer array.

#### [FSPtrArrayRemoveAll](#)

Cleans up the array.

#### [FSPtrArrayRemoveAt](#)

Removes a number of elements at specified position.

#### [FSPtrArraySetAt](#)

Overwrites an element specified by an index number.

#### [FSPtrArraySetAtGrow](#)

Sets an element value at specified position. Potentially grow the array.

#### [FSPtrArraySetSize](#)

Changes the allocated size and the growing amount.

## Functions detail

### FSPtrArrayAdd

#### Syntax

```
FS_INT32 FSPtrArrayAdd (
    FS_PtrArray arr,
    void* newItem
```

);

**Description**

Adds an element at the tail. Potentially grow the array.

**Parameter**

---

arr	[In] The input pointer array.
-----	-------------------------------

---

newItem	[In] The input element.
---------	-------------------------

---

**Return**

The added element's index in the array.

**Head file reference**

fs\_basicTempl.h: 1200

**FSPtrArrayAppend****Syntax**

```
FS_INT32 FSPtrArrayAppend (
    FS_PtrArray arr,
    const FS_PtrArray srcArr
);
```

**Description**

Appends an array.

**Parameter**

---

arr	[In] The input pointer array.
-----	-------------------------------

---

srcArr	[In] The input array.
--------	-----------------------

---

**Return**

The old size in elements.

**Head file reference**

fs\_basicTempl.h: 1210

**FSPtrArrayCopy****Syntax**

```
void FSPtrArrayCopy (
    FS_PtrArray arr,
```

```
FS_PtrArray srcArr  
);
```

**Description**

Copies from an array.

**Parameter**

---

arr	[In] The input pointer array.
-----	-------------------------------

---

srcArr	[In] The input array.
--------	-----------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1220

**FSPtrArrayDestroy****Syntax**

```
void FSPtrArrayDestroy (  
    FS_PtrArray arr  
) ;
```

**Description**

Destroys a pointer array.

**Parameter**

---

arr	[In] The input pointer array.
-----	-------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1121

**FSPtrArrayFind****Syntax**

```
FS_INT32 FSPtrArrayFind (  
    FS_PtrArray arr,  
    void* item,  
    FS_INT32 nstartIndex  
) ;
```

**Description**

Finds an element from specified position to last

**Parameter**

arr	[In] The input pointer array.
item	[In] The input element.
nstartIndex	[In] Specifies the zero-based index of start element to find.

**Return**

An index of the found element. It can be -1 for finding none.

**Head file reference**

fs\_basicTempl.h: 1274

**FSPtrArrayGetAt****Syntax**

```
void* FSPtrArrayGetAt (
    FS_PtrArray arr,
    FS_INT32 index
);
```

**Description**

Retrieves an element specified by an index number.

**Parameter**

arr	[In] The input pointer array.
index	[In] Specifies the zero-based index of the element.

**Return**

An element.

**Head file reference**

fs\_basicTempl.h: 1168

**FSPtrArrayGetDataPtr****Syntax**

```
void** FSPtrArrayGetDataPtr (
    FS\_PtrArray arr,
    FS\_INT32 index
);
```

**Description**

Gets a pointer to the specified element in the array. Direct pointer access.

**Parameter**

---

arr	[In] The input pointer array.
index	[In] Specifies the zero-based index of element in the array.

---

**Return**

A pointer to the specified element.

**Head file reference**

fs\_basicTempl.h: 1230

**FSPtrArrayGetSize****Syntax**

```
FS_INT32 FSPtrArrayGetSize (
    FS\_PtrArray arr
);
```

**Description**

Gets the number of elements in the array.

**Parameter**

---

arr	[In] The input pointer array.
-----	-------------------------------

---

**Return**

The number of elements in the array.

**Head file reference**

fs\_basicTempl.h: 1130

**FSPtrArrayGetUpperBound****Syntax**

```
FS_INT32 FSPtrArrayGetUpperBound (
    FS\_PtrArray arr
);
```

**Description**

Gets the upper bound in the array, actually the maximum valid index.

**Parameter**

---

arr	[In] The input pointer array.
-----	-------------------------------

---

**Return**

The upper bound.

**Head file reference**

fs\_basicTempl.h: 1139

**FSPtrArrayInsertAt****Syntax**

```
void FSPtrArrayInsertAt (
    FS\_PtrArray arr,
    FS\_INT32 index,
    void* newItem,
    FS\_INT32 nCount
);
```

**Description**

Inserts one or more continuous element at specified position.

**Parameter**

---

arr	[In] The input pointer array.
-----	-------------------------------

---

---

index	[In] Specifies the zero-based index in the array.
-------	---

---

---

newItem	[In] Specifies the element value to insert.
---------	---

---

---

nCount	[In] Specifies the count of the element to insert.
--------	--

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1240

**FSPtrArrayInsertNewArrayAt**

**Syntax**

```
void FSPtrArrayInsertNewArrayAt (
    FS\_PtrArray arr,
    FS\_INT32 nstartIndex,
    FS\_PtrArray newArray
);
```

**Description**

Inserts an array at specified position.

**Parameter**


---

arr	[In] The input pointer array.
nstartIndex	[In] Specifies the zero-based index of start element to insert at.
newArray	[In] The input array.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1263

**FSPtrArrayNew****Syntax**

```
FS\_PtrArray FSPtrArrayNew (void );
```

**Description**

Creates a new empty pointer array.

**Return**

A new empty pointer array.

**Head file reference**

fs\_basicTempl.h: 1112

**FSPtrArrayRemoveAll****Syntax**

```
void FSPtrArrayRemoveAll (
    FS\_PtrArray arr
);
```

**Description**

Cleans up the array.

**Parameter**


---

arr	[In] The input pointer array.
-----	-------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1159

**FSPtrArrayRemoveAt****Syntax**

```
void FSPtrArrayRemoveAt (
    FS\_PtrArray arr,
    FS\_INT32 index,
    FS\_INT32 nCount
);
```

**Description**

Removes a number of elements at specified position.

**Parameter**


---

arr	[In] The input pointer array.
-----	-------------------------------

---

index	[In] Specifies the zero-based index in the array.
-------	---

---

nCount	[In] Specifies the count of element to remove.
--------	--

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1252

**FSPtrArraySetAt****Syntax**

```
void FSPtrArraySetAt (
    FS\_PtrArray arr,
    FS\_INT32 index,
    void* newItem
);
```

**Description**

Overwrites an element specified by an index number.

**Parameter**

arr	[In] The input pointer array.
index	[In] Specifies the zero-based index of the element.
newItem	[In] An element

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1178

**FSPtrArraySetAtGrow****Syntax**

```
void FSPtrArraySetAtGrow (
    FS\_PtrArray arr,
    FS\_INT32 index,
    void* newItem
);
```

**Description**

Sets an element value at specified position. Potentially grow the array.

**Parameter**

arr	[In] The input pointer array.
index	[In] Specifies the zero-based index of element in the array.
newItem	[In] The input element.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1189

## FSPtrArraySetSize

### Syntax

```
void FSPtrArraySetSize (
    FS\_PtrArray arr,
    FS\_INT32 nNewSize,
    FS\_INT32 nGrowBy
);
```

### Description

Changes the allocated size and the growing amount.

### Parameter

arr	[In] The input pointer array.
-----	-------------------------------

nNewSize	[In] The new size in elements expected.
----------	---

nGrowBy	[In] The grow amount in elements expected, nGrowBy can be -1 for the grow amount unchanged.
---------	---

### Return

void

### Head file reference

[fs\\_basicTempl.h](#): 1148

## FS\_StreamWriteHandler

### [Return from Used by](#)

### Description

stream writing interface.

### Returned from

### [FSStreamWriteHandlerNew](#)

### Used by

[FPDCreatorCreate3](#)  
[FPDWrapperCreatorCreate](#)  
[FSStreamWriteHandlerDestroy](#)  
[FSStreamWriteHandlerWriteBlock](#)

### Callbacks

## Callbacks summary

### [FSStreamWrite\\_WriteBlock](#)

Write a block data.

### [FSStreamWrite\\_Release](#)

Called when to release everything.

## Callbacks detail

### FSStreamWrite\_WriteBlock

#### Syntax

```
typedef FS_BOOL (*FSStreamWrite_WriteBlock)(  
    FS\_LPVOID clientData,  
    const void* pData,  
    FS\_DWORD size  
)
```

#### Description

Write a block data.

#### Parameter

clientData	[In] The user-supplied data.
------------	------------------------------

pData	[In] The block data.
-------	----------------------

size	[In] The length in bytes of the block data.
------	---

#### Return

TRUE for success, FALSE for failure.

#### Head file reference

fs\_basicExpT.h: 1275

#### Group

[FS\\_StreamWrite](#)

### FSStreamWrite\_Release

#### Syntax

```
typedef void (*FSStreamWrite_Release)(  
    FS\_LPVOID clientData  
)
```

#### Description

Called when to release everything. Called when to release everything

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fs\_basicExpT.h: 1288

**Group**[FS\\_StreamWrite](#)

## Functions

### Functions summary

[\*\*FSStreamWriteHandlerDestroy\*\*](#)

Frees stream write handler.

[\*\*FSStreamWriteHandlerNew\*\*](#)

Creates the stream access handler.

[\*\*FSStreamWriteHandlerWriteBlock\*\*](#)

Write a block data.

### Functions detail

[\*\*FSStreamWriteHandlerDestroy\*\*](#)**Syntax**

```
void FSStreamWriteHandlerDestroy (
    FS\_StreamWriteHandler streamWritehandler
);
```

**Description**

Frees stream write handler.

**Parameter**

---

streamWritehandler	[In] The stream access handler.
--------------------	---------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2642

[\*\*FSStreamWriteHandlerNew\*\*](#)

**Syntax**

```
FS_StreamWriteHandler FSStreamWriteHandlerNew (
    FS\_StreamWrite streamWrite
);
```

**Description**

Creates the stream access handler.

**Parameter**

---

streamWrite	[In] The input stream write callbacks.
-------------	--

---

**Return**

The stream access interface.

**Head file reference**

fs\_basicTempl.h: 2633

**FSStreamWriteHandlerWriteBlock****Syntax**

```
FS_BOOL FSStreamWriteHandlerWriteBlock (
    FS\_StreamWriteHandler streamWritehandler,
    const void* pData,
    FS\_DWORD size
);
```

**Description**

Write a block data.

**Parameter**

---

streamWritehandler	[In] The stream access handler.
--------------------	---------------------------------

---

---

pData	[In] The block data.
-------	----------------------

---

---

size	[In] The length in bytes of the block data.
------	---

---

**Return**

[TRUE](#) for success, [FALSE](#) for failure.

**Head file reference**

fs\_basicTempl.h: 2651

**FS\_UTF8Decoder**

## [Return from Used by](#)

### Description

A UTF8 decoder object. You can use it to decode UTF-8 encoded data. See [FSUTF8DecoderNew](#) , [FSUTF8DecoderDestroy](#) .

### Returned from

#### [FSUTF8DecoderNew](#)

### Used by

[FSUTF8DecoderAppendChar](#)  
[FSUTF8DecoderClear](#)  
[FSUTF8DecoderClearStatus](#)  
[FSUTF8DecoderDestroy](#)  
[FSUTF8DecoderGetResult](#)  
[FSUTF8DecoderInput](#)

### Functions

#### Functions summary

##### [FSUTF8DecoderAppendChar](#)

Appends characters to wide text buffer.

##### [FSUTF8DecoderClear](#)

Clears the decoding status and sets the output wide text buffer to be empty.

##### [FSUTF8DecoderClearStatus](#)

Clears the decoding status.

##### [FSUTF8DecoderDestroy](#)

Destroys the UTF8 decoder object.

##### [FSUTF8DecoderGetResult](#)

Gets the result.

##### [FSUTF8DecoderInput](#)

Inputs a byte.

##### [FSUTF8DecoderNew](#)

Creates a new UTF8 decoder object.

#### Functions detail

##### [FSUTF8DecoderAppendChar](#)

###### Syntax

```
void FSUTF8DecoderAppendChar (
    FS\_UTF8Decoder decoder,
    FS\_DWORD ch
);
```

###### Description

Appends characters to wide text buffer.

**Parameter**

---

decoder	[In] The input UTF8 decoder.
---------	------------------------------

---

ch

[In] The input character.

**Return**

void.

**Head file reference**

fs\_basicTempl.h: 3458

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FSUTF8DecoderClear****Syntax**

```
void FSUTF8DecoderClear (
    FS\_UTF8Decoder decoder
);
```

**Description**

Clears the decoding status and sets the output wide text buffer to be empty.

**Parameter**

---

decoder	[In] The input UTF8 decoder.
---------	------------------------------

---

**Return**

void.

**Head file reference**

fs\_basicTempl.h: 3437

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FSUTF8DecoderClearStatus****Syntax**

```
void FSUTF8DecoderClearStatus (
    FS\_UTF8Decoder decoder
);
```

**Description**

Clears the decoding status.

**Parameter**

---

decoder	[In] The input UTF8 decoder.
---------	------------------------------

---

**Return**

void.

**Head file reference**

fs\_basicTempl.h: 3469

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FSUTF8DecoderDestroy****Syntax**

```
void FSUTF8DecoderDestroy (
    FS\_UTF8Decoder decoder
);
```

**Description**

Destroys the UTF8 decoder object.

**Parameter**

---

decoder	[In] The input UTF8 decoder.
---------	------------------------------

---

**Return**

void.

**Head file reference**

fs\_basicTempl.h: 3427

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FSUTF8DecoderGetResult****Syntax**

```
void FSUTF8DecoderGetResult (
    FS\_UTF8Decoder decoder,
    FS\_WideString* outResult
);
```

**Description**

Gets the result.

**Parameter**

---

decoder	[In] The input UTF8 decoder.
outResult	[Out] It receives the result.

---

**Return**

void.

**Head file reference**

fs\_basicTempl.h: 3479

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FSUTF8DecoderInput****Syntax**

```
void FSUTF8DecoderInput (
    FS\_UTF8Decoder decoder,
    FS\_BYTE byte
);
```

**Description**

Inputs a byte.

**Parameter**

---

decoder	[In] The input UTF8 decoder.
byte	[In] The input byte.

---

**Return**

void.

**Head file reference**

fs\_basicTempl.h: 3447

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FSUTF8DecoderNew**

**Syntax**

```
FS_UTF8Decoder FSUTF8DecoderNew (void );
```

**Description**

Creates a new UTF8 decoder object.

**Return**

The UTF8 decoder object.

**Head file reference**

fs\_basicTempl.h: 3417

**Since**

[SDK LATEEST VERSION > 1.0](#)

## FS\_UTF8Encoder

[Return from Used by](#)

**Description**

A UTF8 encoder object. You can use it to encode data into UTF-8. See [FSUTF8EncoderNew](#) , [FSUTF8EncoderDestroy](#) .

**Returned from**

[FSUTF8EncoderNew](#)

**Used by**

[FSUTF8EncoderAppendStr](#)  
[FSUTF8EncoderDestroy](#)  
[FSUTF8EncoderGetResult](#)  
[FSUTF8EncoderInput](#)

## Functions

**Functions summary**

[FSUTF8EncoderAppendStr](#)

Appends a non-buffered byte string.

[FSUTF8EncoderDestroy](#)

Destroys the UTF8 encoder object.

[FSUTF8EncoderGetResult](#)

Gets the result.

[FSUTF8EncoderInput](#)

Inputs a unicode.

[FSUTF8EncoderIsUTF8Data](#)

Checks whether a data buffer is UTF-8 encoded or not.

**[FSUTF8EncoderNew](#)**

Creates a new UTF8 encoder object.

**Functions detail****[FSUTF8EncoderAppendStr](#)****Syntax**

```
void FSUTF8EncoderAppendStr (
    FS\_UTF8Encoder encoder,
    FS\_ByteString str
);
```

**Description**

Appends a non-buffered byte string.

**Parameter**

encoder	[In] The input UTF8 encoder.
str	[In] A non-buffered byte string.

**Return**

void.

**Head file reference**

[fs\\_basicTempl.h](#): 3529

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**[FSUTF8EncoderDestroy](#)****Syntax**

```
void FSUTF8EncoderDestroy (
    FS\_UTF8Encoder encoder
);
```

**Description**

Destroys the UTF8 encoder object.

**Parameter**

encoder	[In] The input UTF8 encoder.
---------	------------------------------

**Return**

void.

**Head file reference**

fs\_basicTempl.h: 3508

**Since**[SDK LATEEST VERSION > 1.0](#)**FSUTF8EncoderGetResult****Syntax**

```
void FSUTF8EncoderGetResult (
    FS\_UTF8Encoder encoder,
    str
);
```

**Description**

Gets the result.

**Parameter**

---

encoder	[In] The input UTF8 encoder.
---------	------------------------------

---

str	[Out] It receives the result.
-----	-------------------------------

---

**Return**

void.

**Head file reference**

fs\_basicTempl.h: 3540

**Since**[SDK LATEEST VERSION > 1.0](#)**FSUTF8EncoderInput****Syntax**

```
void FSUTF8EncoderInput (
    FS\_UTF8Encoder encoder,
    FS\_WCHAR unicode
);
```

**Description**

Inputs a unicode.

**Parameter**

---

encoder	[In] The input UTF8 encoder.
unicode	[In] The input unicode.

---

**Return**

void.

**Head file reference**

fs\_basicTempl.h: 3518

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FSUTF8EncoderIsUTF8Data****Syntax**

```
FS_BOOL FSUTF8EncoderIsUTF8Data (
    FS_LPCBYTE pData,
    FS_INT32* pLen
);
```

**Description**

Checks whether a data buffer is UTF-8 encoded or not.

**Parameter**


---

pData	[In] The pointer to the data buffer.
pLen	[In/Out] The pointer to the length of the data buffer, in bytes. When this function returns, it stores the number of bytes scanned.

---

**Return**

TRUE if all data is UTF-8 encoded, FALSE if the data is not UTF-8 format..

**Head file reference**

fs\_basicTempl.h: 3551

**Since**[SDK\\_LATEEST\\_VERSION > 7.2.2](#)**FSUTF8EncoderNew****Syntax**

```
FS_UTF8Encoder FSUTF8EncoderNew (void );
```

**Description**

Creates a new UTF8 encoder object.

**Return**

The UTF8 encoder object.

**Head file reference**

fs\_basicTempl.h: 3498

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

# FS\_UUID

## Description

## Functions

### Functions summary

**[FSUUIDGenerate](#)**

Generates a UUID.

**[FSUUIDGenerateTime](#)**

Generates a UUID constructed by time.

**[FSUUIDGenerateRandom](#)**

Generates a UUID randomly.

**[FSUUIDSetTsPath](#)**

Sets the Ts path.

**[FSUUIDSetState](#)**

Sets the last state.

**[FSUUIDSetUserData](#)**

Sets the user data.

### Functions detail

#### **FSUUIDGenerate**

**Syntax**

```
FS_INT32 FSUUIDGenerate (
    FS_ByteString* outID,
    FS_INT32 userData,
    FS_ByteString* outLastState
);
```

**Description**

Generates a UUID.

**Parameter**

---

outID	[Out] It receives the ID.
userData	[In] The input user data.
outLastState	[In/Out] It receives the last state of the ID.

---

**Return**

The type of the UUID. See the definition of [FSUUIDType](#).

**Head file reference**

fs\_basicTempl.h: 3644

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FSUUIDGenerateTime****Syntax**

```
FS_INT32 FSUUIDGenerateTime (
    FS\_ByteString* outID,
    FS\_INT32 userData,
    FS\_ByteString* outLastState,
    FS\_BOOL bOnlyUid
);
```

**Description**

Generates a UUID constructed by time.

**Parameter**


---

outID	[Out] It receives the ID.
userData	[In] The input user data.
outLastState	[In/Out] It receives the last state of the ID.
bOnlyUid	[In] Whether generates the ID only or not.

---

**Return**

The type of the UUID. See the definition of [FSUUIDType](#).

**Head file reference**

fs\_basicTempl.h: 3656

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)**FSUUIDGenerateRandom****Syntax**

```
FS_INT32 FSUUIDGenerateRandom (
    FS_ByteString* outID,
    FS_INT32 userData
);
```

**Description**

Generates a UUID randomly.

**Parameter**

outID	[Out] It receives the ID.
-------	---------------------------

userData	[In] The input user data.
----------	---------------------------

**Return**

The type of the UUID. See the definition of [FSUUIDType](#).

**Head file reference**

fs\_basicTempl.h: 3669

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FSUUIDSetTsPath****Syntax**

```
FS_INT32 FSUUIDSetTsPath (
    FS_ByteString bsTsPath
);
```

**Description**

Sets the Ts path.

**Parameter**

bsTsPath	[In] The input Ts path.
----------	-------------------------

**Return****Head file reference**

fs\_basicTempl.h: 3680

**Since**  
[SDK LATEEST VERSION > 7.3](#)

## FSUUIDSetState

### Syntax

```
FS_INT32 FSUUIDSetState (
    FS ByteString bsLastState
);
```

### Description

Sets the last state.

### Parameter

---

bsLastState	[In] The input last state.
-------------	----------------------------

---

### Return

#### Head file reference

fs\_basicTempl.h: 3690

**Since**

[SDK LATEEST VERSION > 7.3](#)

## FSUUIDSetUserData

### Syntax

```
FS_INT32 FSUUIDSetUserData (
    FS ByteString bsUserData
);
```

### Description

Sets the user data.

### Parameter

---

bsUserData	[In] The input user data.
------------	---------------------------

---

### Return

#### Head file reference

fs\_basicTempl.h: 3700

**Since**

[SDK LATEEST VERSION > 7.3](#)

# FS\_WideString

## [Return from Used by](#)

### Description

An object used to store unicode characters. On Windows platforms, a wide string is represented by UTF-16LE encoding; On Unix/Linux platforms, a wide string is represented by UCS-4 encoding. See [FSWideStringNew](#) , [FSWideStringNew2](#) , [FSWideStringNew3](#) , [FSWideStringDestroy](#) .

### Returned from

[FSWideStringArrayNew](#)  
[FSWideStringNew](#)  
[FSWideStringNew2](#)  
[FSWideStringNew3](#)

### Used by

[FSBase64EncoderEncode2](#)  
[FSByteStringConvertFrom](#)  
[FSByteStringUTF8Decode](#)  
[FSCodeTransformationDecodeText](#)  
[FSCodeTransformationDecodeText2](#)  
[FSUTF8DecoderGetResult](#)  
[FSWideStringArrayAdd](#)  
[FSWideStringArrayDestroy](#)  
[FSWideStringArrayGetAt](#)  
[FSWideStringArrayGetSize](#)  
[FSWideStringArrayRemoveAll](#)  
[FSWideStringArrayRemoveAt](#)  
[FSXMLElementAddChildContent](#)  
[FSXMLElementGetAttrByIndex](#)  
[FSXMLElementGetAttributeValue](#)  
[FSXMLElementGetAttributeValue2](#)  
[FSXMLElementGetContent](#)  
[FSXMLElementInsertChildContent](#)  
[FSXMLElementSetAttrValue](#)  
[FRAppAddLog](#)  
[FRAppCommandLineGetArgument](#)  
[FRAppCommandLineGetSafeArgument](#)  
[FRAppCreateAnEmptyFrameWnd](#)  
[FRAppGetAppDataPath](#)  
[FRAppGetAppName](#)  
[FRAppGetKeyfileStartAndExpireDate](#)  
[FRAppGetLocalFontName](#)  
[FRAppGetModuleFileName](#)  
[FRAppGetRecentFileList](#)  
[FRAppGetStatusBarBkGroundPath](#)  
[FRAppGetVersion](#)  
[FRAppRegistryGetEntryName](#)

[FRAppRegistryGetKeyName](#)  
[FRAppRegistryGetProfilePath](#)  
[FRAppRegistryGetString](#)  
[FRDocGenerateRedactions](#)  
[FRDocGetCreateDocSourceFileName](#)  
[FRDocGetCreateDocSourceFilePath](#)  
[FRDocGetCurrentSecurityMethodName](#)  
[FRDocGetDocShowTitle](#)  
[FRDocGetFilePath](#)  
[FRDocViewGetLRCViewText](#)  
[FRDocViewGetTagDocViewText](#)  
[FRFormatToolsCleanOwnerFontNameArr](#)  
[FRFormatToolsGetFontFaceName](#)  
[FRFormatToolsGetFontListItem](#)  
[FRFormatToolsGetFontName](#)  
[FRFormatToolsSetOwnerFontNameArr](#)  
[FRFuncBtnGetToolTip](#)  
[FRLanguageFormatEx](#)  
[FRLanguageGetLocalLangName](#)  
[FRLanguageJSPluginGetMessage](#)  
[FRLanguageLoadVersionRes](#)  
[FRMenuItemGetTitle](#)  
[FRMessageBarSetButtonHelpText](#)  
[FRMessageBarSetText](#)  
[FRRibbonCategoryGetContextTitle](#)  
[FRRibbonCategoryGetKey](#)  
[FRRibbonCategoryGetTitle](#)  
[FRRibbonColorButtonGetAutomaticButtonLabel](#)  
[FRRibbonColorButtonGetAutomaticButtonToolTip](#)  
[FRRibbonColorButtonGetGroupLabel](#)  
[FRRibbonColorButtonGetOtherButtonLabel](#)  
[FRRibbonColorButtonGetOtherButtonToolTip](#)  
[FRRibbonComboBoxGetEditText](#)  
[FRRibbonComboBoxGetItem](#)  
[FRRibbonEditGetText](#)  
[FRRibbonElementGetDescription](#)  
[FRRibbonElementGetKey](#)  
[FRRibbonElementGetTitle](#)  
[FRRibbonElementGetTooltip](#)  
[FRRibbonFontComboBoxGetEditText](#)  
[FRRibbonFontComboBoxGetFontName](#)  
[FRRibbonFontComboBoxGetItem](#)  
[FRRibbonFontComboBoxGetScriptName](#)  
[FRRibbonListButtonAddGroup](#)  
[FRRibbonListButtonGetGroupTitle](#)  
[FRRibbonListButtonGetItemTitle](#)  
[FRRibbonListButtonGetItemToolTip](#)  
[FRRibbonPaletteButtonGetGroupTitle](#)  
[FRRibbonPaletteButtonGetItemToolTip](#)  
[FRRibbonPaletteButtonGetMenuButtonTooltip](#)  
[FRRibbonPaletteButtonGetScrollBarTooltip](#)  
[FRRibbonPaletteButtonSetItemAccNameTitle](#)  
[FRRibbonPanelGetTitle](#)  
[FRSpellCheckCheckString](#)  
[FRSpellCheckSuggestWords](#)  
[FRStatusBarGetComboBoxPageText](#)  
[FRStatusBarGetBkGroundPath](#)

[FRSysGetSkinNameByIndex](#)  
[FRTTextSelectToolGetSelectedText](#)  
[FRToolBarGetTitle](#)  
[FRToolBarGetLabelText](#)  
[FPDFDFDocGetWin32Path](#)  
[FPDACTIONGetFilePath](#)  
[FPDACTIONGetJavaScript](#)  
[FPDBookmarkGetTitle](#)  
[FPDDictionaryGetUnicodeText](#)  
[FPDFFileSpecGetFileName](#)  
[FPDFFontDecodeString](#)  
[FPDFFontFXFontGetPsName](#)  
[FPDFFontUnicodeFromCharCode](#)  
[FPDFFontUnicodeFromCharCodeEx](#)  
[FPDFFormControlGetDownCaption](#)  
[FPDFFormControlGetExportValue](#)  
[FPDFFormControlGetNormalCaption](#)  
[FPDFFormControlGetRolloverCaption](#)  
[FPDFFormFieldGetAlternateName](#)  
[FPDFFormFieldGetDefaultValue](#)  
[FPDFFormFieldGetFullName](#)  
[FPDFFormFieldGetMappingName](#)  
[FPDFFormFieldGetOptionLabel](#)  
[FPDFFormFieldGetOptionValue](#)  
[FPDFFormFieldGetRichTextString](#)  
[FPDFFormFieldGetValue](#)  
[FPDFXFontEncodingUnicodeFromCharCode](#)  
[FPDIInterForm GetAllFieldNames](#)  
[FPDIInterForm NewControl](#)  
[FPDIInterForm NewField](#)  
[FPDIInterForm RenameControl](#)  
[FPDIInterForm RenameField](#)  
[FPDIInterForm RenameField2](#)  
[FPDIInterForm ValidateFieldName](#)  
[FPDIInterForm ValidateFieldName2](#)  
[FPDIInterForm ValidateFieldName3](#)  
[FPDLINKExtractGetURL](#)  
[FPDNameTreeLookupValue](#)  
[FPDOBJArchiveLoaderLoadWideString](#)  
[FPDOBJArchiveSaverSaveWideString](#)  
[FPDOBJECTGetUnicodeText](#)  
[FPDOCGroupGetCreatorInfo](#)  
[FPDOCGroupGetName](#)  
[FPDOCGroup GetUserType](#)  
[FPDOCGroupSetGetSubGroupSetName](#)  
[FPDPAGEGetPageText\\_Uncode](#)  
[FPDPARSERGetUnicodePassword](#)  
[FPDRENDITIONGetFloatingWindowTitle](#)  
[FPDRENDITIONGetMediaClipName](#)  
[FPDRENDITIONGetMediaDescriptions](#)  
[FPDRENDITIONGetRenditionName](#)  
[FPDRENDITIONSetFloatingWindowTitle](#)  
[FPDTXTPAGEGetPageText](#)  
[FPDTXTPAGEGetTextByRect](#)  
[FSWideStringCastToLPCWSTR](#)  
[FSWideStringCompare](#)  
[FSWideStringCompare2](#)

---

[\*\*FSWideStringCompareNoCase\*\*](#)  
[\*\*FSWideStringCompareNoCase2\*\*](#)  
[\*\*FSWideStringConcat\*\*](#)  
[\*\*FSWideStringConcat2\*\*](#)  
[\*\*FSWideStringConvertFrom\*\*](#)  
[\*\*FSWideStringCopy\*\*](#)  
[\*\*FSWideStringDelete\*\*](#)  
[\*\*FSWideStringDestroy\*\*](#)  
[\*\*FSWideStringEmpty\*\*](#)  
[\*\*FSWideStringEqual\*\*](#)  
[\*\*FSWideStringEqual2\*\*](#)  
[\*\*FSWideStringFill\*\*](#)  
[\*\*FSWideStringFind\*\*](#)  
[\*\*FSWideStringFind2\*\*](#)  
[\*\*FSWideStringFormat\*\*](#)  
[\*\*FSWideStringFormatV\*\*](#)  
[\*\*FSWideStringFromLocal\*\*](#)  
[\*\*FSWideStringFromLocal2\*\*](#)  
[\*\*FSWideStringFromUTF16LE\*\*](#)  
[\*\*FSWideStringFromUTF8\*\*](#)  
[\*\*FSWideStringGetAt\*\*](#)  
[\*\*FSWideStringGetInteger\*\*](#)  
[\*\*FSWideStringGetLength\*\*](#)  
[\*\*FSWideStringInsert\*\*](#)  
[\*\*FSWideStringIsEmpty\*\*](#)  
[\*\*FSWideStringLeft\*\*](#)  
[\*\*FSWideStringMakeLower\*\*](#)  
[\*\*FSWideStringMakeUpper\*\*](#)  
[\*\*FSWideStringMid\*\*](#)  
[\*\*FSWideStringMid2\*\*](#)  
[\*\*FSWideStringRemove\*\*](#)  
[\*\*FSWideStringReplace\*\*](#)  
[\*\*FSWideStringReserve\*\*](#)  
[\*\*FSWideStringRight\*\*](#)  
[\*\*FSWideStringSetAt\*\*](#)  
[\*\*FSWideStringTrimLeft\*\*](#)  
[\*\*FSWideStringTrimLeft2\*\*](#)  
[\*\*FSWideStringTrimLeft3\*\*](#)  
[\*\*FSWideStringTrimRight\*\*](#)  
[\*\*FSWideStringTrimRight2\*\*](#)  
[\*\*FSWideStringTrimRight3\*\*](#)  
[\*\*FSWideStringUTF16LE\\_Encode\*\*](#)  
[\*\*FSWideStringUTF8Encode\*\*](#)

## Functions

### Functions summary

#### [\*\*FSWideStringCastToLPCWSTR\*\*](#)

Cast the wide string to wide char typed pointer.

#### [\*\*FSWideStringCompare\*\*](#)

Compares the the string with another FS\_WCHAR buffer. case-sensitive.

#### [\*\*FSWideStringCompare2\*\*](#)

Compares the the string with another string. case-sensitive.

#### [\*\*FSWideStringCompareNoCase\*\*](#)

Compares the string with a wide character string, case-insensitive.

**FSWideStringCompareNoCase2**

Compares the the string with another, case-insensitive.

**FSWideStringConcat**

Concatenates a source byte string.

**FSWideStringConcat2**

Concatenates a normal unicode string to wide string.

**FSWideStringConvertFrom**

Loads MBCS data into this wide string, using specified character mapper. If no character mapper specified, the system default mapper will be used.

**FSWideStringCopy**

Copies from a source byte string.

**FSWideStringDelete**

Deletes one or more characters starting from specific position.

**FSWideStringDestroy**

Destroys the wide string.

**FSWideStringEmpty**

Sets this string to be empty.

**FSWideStringEqual**

Checks if this string equals to FS\_WCHAR buffer.

**FSWideStringEqual2**

Checks if this string equals to another.

**FSWideStringFill**

Fills a normal unicode string to wide string.

**FSWideStringFind**

Finds a sub-string, from specific position. Only first occurrence is found.

**FSWideStringFind2**

Finds a character, from specific position. Only first occurrence is found.

**FSWideStringFormat**

Formats a number of parameters into this byte string.

**FSWideStringFormatV**

Formats a number of parameters into this wide string, using va\_list.

**FSWideStringFromLocal**

Creates a wide string from system multi-byte charset.

**FSWideStringFromLocal2**

Creates a wide string from system multi-byte charset.

**FSWideStringFromUTF16LE**

Creates a wide string from UTF16LE encoded string.

**FSWideStringFromUTF8**

Creates a wide string from UTF-8 string (ASCII string compatible).

**FSWideStringGetAt**

Retrieves a single wide char specified by an index number.

**FSWideStringGetInteger**

Converts to other data type.

**FSWideStringGetLength**

Gets number of bytes in the byte string (not counting any possible terminator).

**FSWideStringInsert**

Inserts a character before specific position.

**FSWideStringIsEmpty**

Determines whether it is empty or not.

**FSWideStringLeft**

Extracts the leftmost count bytes as a substring.

**FSWideStringMakeLower**

Changes case of English letters to lower.

**FSWideStringMakeUpper**

Changes case of English letters to upper.

**FSWideStringMid**

Extracts a substring starting at position nFirst (zero-based) to last.

**FSWideStringMid2**

Extracts a substring starting at position nFirst (zero-based) to position nFirst + count

**FSWideStringNew**

Creates a new empty wide string.

**FSWideStringNew2**

Creates a new wide string from a single wide character.

**FSWideStringNew3**

Creates a new wide string from a wide character string.

**FSWideStringRemove**

Removes all occurrence of a particular character.

**FSWideStringReplace**

Replace all patterns in the string with a new sub-string.

**FSWideStringReserve**

Reserves a buffer that can hold specific number of bytes.

**FSWideStringRight**

Extracts the rightmost count as a substring.

**FSWideStringSetAt**

Overwrites a single byte specified by an index number.

**FSWideStringTrimLeft**

Trims white spaces from the left side of the byte string.

**FSWideStringTrimLeft2**

Trims continuous occurrences of specified characters from left side of the byte string.

**FSWideStringTrimLeft3**

Trims continuous occurrences of specified characters from left side of the byte string.

**FSWideStringTrimRight**

Trims white spaces from the right side of the byte string.

**FSWideStringTrimRight2**

Trims continuous occurrences of specified character from right side of the byte string.

**FSWideStringTrimRight3**

Trims continuous occurrences of specified characters from right side of the byte string.

**FSWideStringUTF16LE\_Encode**

Does UTF16LE encoding. Gets UTF-16LE encoded memory block.

**FSWideStringUTF8Encode**

Does UTF8 encoding.

## Functions detail

### FSWideStringCastToLPCWSTR

#### Syntax

```
FS_LPCWSTR FSWideStringCastToLPCWSTR (
    FS_WideString wstr
);
```

**Description**

Cast the wide string to wide char typed pointer.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

**Return**

A wide char buffer.

**Head file reference**

fs\_stringTempl.h: 1015

**FSWideStringCompare****Syntax**

```
FS_INT32 FSWideStringCompare (
    FS\_WideString wstr1,
    FS\_LPCWSTR wstr2
);
```

**Description**

Compares the the string with another FS\_WCHAR buffer, case-sensitive.

**Parameter**

---

wstr1	[In] The input wide string.
-------	-----------------------------

---

---

wstr2	[In] The FS_WCHAR buffer to be compared
-------	---

---

**Return**

-1 if this string is "smaller" (in alphabetic order) than the other, 0 for equal, 1 for larger in alphabetic order.

**Head file reference**

fs\_stringTempl.h: 633

**FSWideStringCompare2****Syntax**

```
FS_INT32 FSWideStringCompare2 (
    FS\_WideString wstr1,
    const FS\_WideString wstr2
);
```

**Description**

Compares the the string with another string, case-sensitive.

**Parameter**


---

wstr1	[In] The input wide string.
wstr2	[In] The byte string to be compared.

---

**Return**

-1 if this string is "smaller" (in alphabetic order) than the other, 0 for equal, 1 for larger in alphabetic order.

**Head file reference**

fs\_stringTempl.h: 644

**FSWideStringCompareNoCase****Syntax**

```
FS_INT32 FSWideStringCompareNoCase (
    FS\_WideString wstr1,
    FS\_LPCWSTR wstr2
);
```

**Description**

Compares the string with a wide character string, case-insensitive.

**Parameter**


---

wstr1	[In] The input wide string.
wstr2	[In] The wide character string to be compared.

---

**Return**

-1 if this string is "smaller" (in alphabetic order) than the other, 0 for equal, 1 for larger in alphabetic order.

**Head file reference**

fs\_stringTempl.h: 1056

**FSWideStringCompareNoCase2****Syntax**

```
FS_INT32 FSWideStringCompareNoCase2 (
    FS\_WideString wstr1,
    const FS\_WideString wstr2
);
```

**Description**

Compares the the string with another. case-insensitive.

**Parameter**

---

wstr1	[In] The input wide string.
-------	-----------------------------

---

wstr2	[In] The wide string to be compared.
-------	--------------------------------------

---

**Return**

-1 if this string is "smaller" (in alphabetic order) than the other, 0 for equal, 1 for larger in alphabetic order.

**Head file reference**

fs\_stringTempl.h: 1067

**FSWideStringConcat****Syntax**

```
void FSWideStringConcat (
    FS\_WideString wstr,
    const FS\_WideString src
);
```

**Description**

Concatenates a source byte string.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

src	[In] The source byte string.
-----	------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 695

**FSWideStringConcat2****Syntax**

```
void FSWideStringConcat2 (
    FS\_WideString wstr,
```

```
FS_LPCWSTR src  
);
```

**Description**

Concatenates a normal unicode string to wide string.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

src
-----

[In] The source normal unicode string.

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 705

**FSWideStringConvertFrom****Syntax**

```
void FSWideStringConvertFrom (  
    FS_WideString wstr,  
    FS_ByteString str,  
    FS_CharMap pCharMap  
)
```

**Description**

Loads MBCS data into this wide string, using specified character mapper. If no character mapper specified, the system default mapper will be used.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

str
-----

[In] A byte string.

---

pCharMap
----------

[In] A character mapper. Invokes  
[FSCharMapGetDefaultMapper](#) to get the default mapper.

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 1078

**Since**

[SDK LATEEST VERSION > 7.1.0.0](#)

**FSWideStringCopy****Syntax**

```
void FSWideStringCopy (
    FS\_WideString wstr,
    const FS\_WideString src
);
```

**Description**

Copies from a source byte string.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

src	[In] The source byte string.
-----	------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 675

**FSWideStringDelete****Syntax**

```
FS_StrSize FSWideStringDelete (
    FS\_WideString wstr,
    FS\_StrSize nIndex,
    FS\_StrSize count
);
```

**Description**

Deletes one or more characters starting from specific position.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

nIndex	[In] Specifies the zero-based index in the byte string for starting deleting.
--------	---

---

---

count	[In] Count of bytes to be deleted.
-------	------------------------------------

---

**Return**

The new length of the byte string.

**Head file reference**

fs\_stringTempl.h: 756

**FSWideStringDestroy****Syntax**

```
void FSWideStringDestroy (
    FS\_WideString wstr
);
```

**Description**

Destroys the wide string.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 606

**FSWideStringEmpty****Syntax**

```
void FSWideStringEmpty (
    FS\_WideString wstr
);
```

**Description**

Sets this string to be empty.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 715

## FSWideStringEqual

### Syntax

```
FS_BOOL FSWideStringEqual (
    FS_WideString wstr1,
    FS_LPCWSTR wstr2
);
```

### Description

Checks if this string equals to FS\_WCHAR buffer.

### Parameter

---

wstr1	[In] The input wide string.
-------	-----------------------------

---

wstr2	[In] The FS_WCHAR buffer to be compared.
-------	--

---

### Return

[TRUE](#) means equal, otherwise not equal.

### Head file reference

fs\_stringTempl.h: 655

**Note:** It's faster than Compare if you just want to check whether two strings equal.

## FSWideStringEqual2

### Syntax

```
FS_BOOL FSWideStringEqual2 (
    FS_WideString wstr1,
    const FS_WideString wstr2
);
```

### Description

Checks if this string equals to another.

### Parameter

---

wstr1	[In] The input wide string.
-------	-----------------------------

---

wstr2	[In] The byte string to be compared.
-------	--------------------------------------

---

### Return

[TRUE](#) means equal, otherwise not equal.

#### Head file reference

fs\_stringTempl.h: 665

**Note:**It's faster than Compare if you just want to check whether two strings equal.

### FSWideStringFill

#### Syntax

```
void FSWideStringFill (
    FS\_WideString wstr,
    FS\_LPCWSTR src
);
```

#### Description

Fills a normal unicode string to wide string.

#### Parameter

---

wstr	[In] The input wide string.
------	-----------------------------

---

src	[In] The source normal unicode string.
-----	--

---

#### Return

void

#### Head file reference

fs\_stringTempl.h: 685

### FSWideStringFind

#### Syntax

```
FS_StrSize FSWideStringFind (
    FS\_WideString wstr,
    const FS\_WideString wstrSub,
    FS\_StrSize start
);
```

#### Description

Finds a sub-string, from specific position. Only first occurrence is found.

#### Parameter

---

wstr	[In] The input wide string.
------	-----------------------------

---

---

wstrSub	[In] The sub-string to be found.
start	[In] Specifies the zero-based index of the starting position to do finding.

---

**Return**

-1:Not found. other value: Specifies position in the string.

**Head file reference**

fs\_stringTempl.h: 842

**FSWideStringFind2****Syntax**

```
FS_StrSize FSWideStringFind2 (
    FS_WideString wstr,
    FS_WCHAR wch,
    FS_StrSize start
);
```

**Description**

Finds a character, from specific position. Only first occurrence is found.

**Parameter**


---

wstr	[In] The input wide string.
wch	[In] The character to be found.

---

start	[In] Specifies the zero-based index of the starting position to do finding.
-------	---

---

**Return**

-1: Not found. other value: Specifies position in the string.

**Head file reference**

fs\_stringTempl.h: 854

**FSWideStringFormat****Syntax**

```
void FSWideStringFormat (
    FS_WideString wstr,
    FS_LPCWSTR lpszFormat,
    ...
);
```

**Description**

Formats a number of parameters into this byte string.

**Parameter**

wstr	[In] The input wide string.
lpszFormat	[In] Specifies a format-control string.
...	[In] arguments list.

**Return**

void

**Head file reference**

fs\_stringTempl.h: 767

**Note:** On desktop platforms, this function supports all the sprintf() formats. On embedded platforms, it supports only a subset of formats:

- Supported types: d, u, f, g, x, X, s, c, %.
- 
- Width field supported;
- 
- Precision not supported;
- 
- Flags supported: '0';

**FSWideStringFormatV****Syntax**

```
void FSWideStringFormatV (
    FS_WideString wstr,
    FS_LPCWSTR lpszFormat,
    va_list argList
);
```

**Description**

Formats a number of parameters into this wide string, using va\_list.

**Parameter**


---

wstr	[In] The input wide string.
lpszFormat	[In] Specifies a format-control string.
argList	[In] Variable-argument lists.

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 1024

**FSWideStringFromLocal****Syntax**

```
void FSWideStringFromLocal (
    FS\_LPSTR str,
    FS\_StrSize len,
    FS\_WideString* outWstr
);
```

**Description**

Creates a wide string from system multi-byte charset.

**Parameter**


---

str	[In] A multi-byte charset string.
len	[In] The length in bytes of the multi-byte charset string. len can be -1 for zero terminated multi-byte charset string.
outWstr	[Out] A wide string.

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 1035

**FSWideStringFromLocal2**

**Syntax**

```
void FSWideStringFromLocal2 (
    FS\_ByteString str,
    FS\_WideString* outWstr
);
```

**Description**

Creates a wide string from system multi-byte charset.

**Parameter**

---

str	[In] A multi-byte charset string.
-----	-----------------------------------

---

outWstr	[Out] A wide string.
---------	----------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 1046

**FSWideStringFromUTF16LE****Syntax**

```
void FSWideStringFromUTF16LE (
    FS\_LPWSTR wstr,
    FS\_StrSize len,
    FS\_WideString* outWstr
);
```

**Description**

Creates a wide string from UTF16LE encoded string.

**Parameter**

---

wstr	[In] A UTF16LE encoded string.
------	--------------------------------

---

len	[In] The length in bytes of the UTF16LE encoded string. len can be -1 for zero terminated UTF16LE string.
-----	---

---

outWstr	[Out] A wide string.
---------	----------------------

---

**Return**

void

**Head file reference**

---

fs\_stringTempl.h: 983

**Note:** len is number of bytes.

### FSWideStringFromUTF8

#### Syntax

```
void FSWideStringFromUTF8 (
    FS\_LPSTR wstr,
    FS\_StrSize len,
    FS\_WideString* outWstr
);
```

#### Description

Creates a wide string from UTF-8 string (ASCII string compatible).

#### Parameter

wstr	[In] A UTF8 string.
len	[In] The length in bytes of the UTF8 string. len can be -1 for zero terminated UTF8 string.
outWstr	[Out] A wide string.

#### Return

void

#### Head file reference

fs\_stringTempl.h: 972

### FSWideStringGetAt

#### Syntax

```
FS_WCHAR FSWideStringGetAt (
    FS\_WideString wstr,
    FS\_StrSize nIndex
);
```

#### Description

Retrieves a single wide char specified by an index number.

#### Parameter

wstr	[In] The input wide string.
------	-----------------------------

---

nIndex	[In] Specifies the zero-based index in the byte string.
--------	---

---

**Return**

A single byte.

**Head file reference**

fs\_stringTempl.h: 724

**FSWideStringGetInteger****Syntax**

```
FS_INT32 FSWideStringGetInteger (
    FS\_WideString wstr
);
```

**Description**

Converts to other data type.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

**Return**

A decimal number

**Head file reference**

fs\_stringTempl.h: 963

**FSWideStringGetLength****Syntax**

```
FS_StrSize FSWideStringGetLength (
    FS\_WideString wstr
);
```

**Description**

Gets number of bytes in the byte string (not counting any possible terminator).

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

**Return**

The Length of the byte string.

**Head file reference**

fs\_stringTempl.h: 615

## FSWideStringInsert

### Syntax

```
FS_StrSize FSWideStringInsert (
    FS\_WideString wstr,
    FS\_StrSize nIndex,
    FS\_WCHAR wch
);
```

### Description

Inserts a character before specific position.

### Parameter

wstr	[In] The input wide string.
nIndex	[In] Specifies the zero-based index in the byte string.
wch	[In] A single character.

### Return

The new length of the byte string.

### Head file reference

fs\_stringTempl.h: 745

## FSWideStringIsEmpty

### Syntax

```
FS_BOOL FSWideStringIsEmpty (
    FS\_WideString wstr
);
```

### Description

Determines whether it is empty or not.

### Parameter

wstr	[In] The input wide string.
------	-----------------------------

### Return

[TRUE](#) means empty, otherwise not empty.

**Head file reference**

fs\_stringTempl.h: 624

**FSWideStringLeft****Syntax**

```
void FSWideStringLeft (
    FS\_WideString wstr,
    FS\_StrSize count,
    FS\_WideString* outStr
);
```

**Description**

Extracts the leftmost count bytes as a substring.

**Parameter**

wstr	[In] The input wide string.
count	[In] The count of bytes expected to extract for the substring.
outStr	[In] Return A leftmost substring.

**Return**

void

**Head file reference**

fs\_stringTempl.h: 820

**FSWideStringMakeLower****Syntax**

```
void FSWideStringMakeLower (
    FS\_WideString wstr
);
```

**Description**

Changes case of English letters to lower.

**Parameter**

wstr	[In] The input wide string.
------	-----------------------------

**Return**

void

**Head file reference**

fs\_stringTempl.h: 866

**FSWideStringMakeUpper****Syntax**

```
void FSWideStringMakeUpper (
    FS\_WideString wstr
);
```

**Description**

Changes case of English letters to upper.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 875

**FSWideStringMid****Syntax**

```
void FSWideStringMid (
    FS\_WideString wstr,
    FS\_StrSize first,
    FS\_WideString* outStr
);
```

**Description**

Extracts a substring starting at position nFirst (zero-based) to last.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

first	[In] Specifies the zero-based index of the starting position in the byte string.
-------	--

---

outStr	[Out] A substring.
--------	--------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 797

**FSWideStringMid2****Syntax**

```
void FSWideStringMid2 (
    FS_WideString wstr,
    FS_StrSize first,
    FS_StrSize count,
    FS_WideString* outStr
);
```

**Description**

Extracts a substring starting at position nFirst (zero-based) to position nFirst + count

**Parameter**

wstr	[In] The input wide string.
first	[In] Specifies the zero-based index of the starting position in the byte string.
count	[In] The count of bytes expected to extract for the sub-string.
outStr	[In] A substring.

**Return**

void

**Head file reference**

fs\_stringTempl.h: 808

**FSWideStringNew****Syntax**

```
FS_WideString FSWideStringNew (void );
```

**Description**

Creates a new empty wide string.

**Return**

A new empty wide string.

**Head file reference**

fs\_stringTempl.h: 578

**FSWideStringNew2****Syntax**

```
FS_WideString FSWideStringNew2 (
    FS_WCHAR wch
);
```

**Description**

Creates a new wide string from a single wide character.

**Parameter**

---

wch	[In] A single character.
-----	--------------------------

---

**Return**

A new wide string.

**Head file reference**

fs\_stringTempl.h: 587

**FSWideStringNew3****Syntax**

```
FS_WideString FSWideStringNew3 (
    FS_LPCWSTR wsz,
    FS_StrSize len
);
```

**Description**

Creates a new wide string from a wide character string.

**Parameter**

---

wsz	[In] Pointer to a character string
-----	------------------------------------

---

len	[In] The length of the character string. len can be -1 for zero terminated string.
-----	--

---

**Return**

A new wide string.

**Head file reference**

fs\_stringTempl.h: 596

### FSWideStringRemove

#### Syntax

```
FS_StrSize FSWideStringRemove (
    FS_WideString wstr,
    FS_WCHAR wch
);
```

#### Description

Removes all occurrence of a particular character.

#### Parameter

---

wstr	[In] The input wide string.
------	-----------------------------

---

wch	[In] Specified the character to be removed.
-----	---

---

#### Return

The number of characters removed.

#### Head file reference

fs\_stringTempl.h: 953

### FSWideStringReplace

#### Syntax

```
FS_StrSize FSWideStringReplace (
    FS_WideString wstr,
    const FS_WideString wstrOld,
    const FS_WideString wstrNew
);
```

#### Description

Replace all patterns in the string with a new sub-string.

#### Parameter

---

wstr	[In] The input wide string.
------	-----------------------------

---

wstrOld	[In] Specified the string to be matched and replaced in the byte string.
---------	--

---

wstrNew	[In] Specified the string to replace.
---------	---------------------------------------

---

**Return**

The number of replaced patterns.

**Head file reference**

fs\_stringTempl.h: 942

**FSWideStringReserve****Syntax**

```
void FSWideStringReserve (
    FS\_WideString wstr,
    FS\_StrSize len
);
```

**Description**

Reserves a buffer that can hold specific number of bytes.

**Parameter**

---

wstr	[In] The input wide string.
len	[In] The Length expected to reserve.

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 785

**Note:** The content of this string won't be changed. This can be used if application anticipates the string may grow many times, in this case, reserving a larger buffer will support string growth without buffer reallocation.

**FSWideStringRight****Syntax**

```
void FSWideStringRight (
    FS\_WideString wstr,
    FS\_StrSize count,
    FS\_WideString* outStr
);
```

**Description**

Extracts the rightmost count as a substring.

**Parameter**

---

wstr	[In] The input wide string.
count	[In] The count of bytes expected to extract for the substring
outStr	[In] Return A rightmost substring.

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 831

**FSWideStringSetAt****Syntax**

```
void FSWideStringSetAt (
    FS\_WideString wstr,
    FS\_StrSize nIndex,
    FS\_WCHAR wch
);
```

**Description**

Overwrites a single byte specified by an index number.

**Parameter**


---

wstr	[In] The input wide string.
nIndex	[In] Specifies the zero-based index in the byte string.
wch	[In] A single character.

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 734

**FSWideStringTrimLeft****Syntax**

```
void FSWideStringTrimLeft (
    FS\_WideString wstr
);
```

**Description**

Trims white spaces from the left side of the byte string.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 913

**FSWideStringTrimLeft2****Syntax**

```
void FSWideStringTrimLeft2 (
    FS\_WideString wstr,
    FS\_WCHAR wchTarget
);
```

**Description**

Trims continuous occurrences of specified characters from left side of the byte string.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

---

wchTarget	[In] The specified character.
-----------	-------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 922

**FSWideStringTrimLeft3****Syntax**

```
void FSWideStringTrimLeft3 (
    FS\_WideString wstr,
    FS\_LPCWSTR wszTargets
);
```

**Description**

Trims continuous occurrences of specified characters from left side of the byte string.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

wszTargets	[In] The specified characters.
------------	--------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 932

**FSWideStringTrimRight****Syntax**

```
void FSWideStringTrimRight (
    FS\_WideString wstr
);
```

**Description**

Trims white spaces from the right side of the byte string.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 884

**FSWideStringTrimRight2****Syntax**

```
void FSWideStringTrimRight2 (
    FS\_WideString wstr,
    FS\_WCHAR wchTarget
);
```

**Description**

Trims continuous occurrences of specified character from right side of the byte string.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

wchTarget	[In] The specified character.
-----------	-------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 893

**FSWideStringTrimRight3****Syntax**

```
void FSWideStringTrimRight3 (
    FS\_WideString wstr,
    FS\_LPCWSTR wszTargets
);
```

**Description**

Trims continuous occurrences of specified characters from right side of the byte string.

**Parameter**


---

wstr	[In] The input wide string.
------	-----------------------------

---

wszTargets	[In] The specified characters.
------------	--------------------------------

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 903

**FSWideStringUTF16LE\_Encode****Syntax**

```
void FSWideStringUTF16LE_Encode (
    FS\_WideString wstr,
    FS\_BOOL bTerminate,
    FS\_ByteString* outEncode
);
```

**Description**

Does UTF16LE encoding. Gets UTF-16LE encoded memory block.

**Parameter**

---

wstr	[In] The input wide string.
bTerminate	[In] need to add terminate symbol? In most of times you can choose 'TRUE'
outEncode	[In] A byte string result.

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 1004

**FSWideStringUTF8Encode****Syntax**

```
void FSWideStringUTF8Encode (
    FS\_WideString wstrSrc,
    FS\_ByteString* outEncode
);
```

**Description**

Does UTF8 encoding.

**Parameter**


---

wstrSrc	[In] The input wide string.
outEncode	[Out] A byte string result.

---

**Return**

void

**Head file reference**

fs\_stringTempl.h: 994

**FS\_WideStringArray****Return from Used by****Description**A wide string array. See [FSWideStringArrayNew](#) , [FSWideStringArrayDestroy](#) .

Returned from

[FSWideStringArrayNew](#)

Used by

[FRAppGetRecentFileList](#)  
[FRFormatToolsCleanOwnerFontNameArr](#)  
[FRFormatToolsSetOwnerFontNameArr](#)  
[FRRibbonListButtonAddGroup](#)  
[FRSpellCheckCheckString](#)  
[FRSpellCheckSuggestWords](#)  
[FPDInterFormGetAllFieldNames](#)  
[FPDOCGroup GetUserType](#)  
[FPDPPageGetPageText Unicode](#)  
[FPDRenditionGetFloatingWindowTitle](#)  
[FPDRenditionGetMediaDescriptions](#)  
[FPDRenditionSetFloatingWindowTitle](#)  
[FSWideStringArrayAdd](#)  
[FSWideStringArrayDestroy](#)  
[FSWideStringArrayGetAt](#)  
[FSWideStringArrayGetSize](#)  
[FSWideStringArrayRemoveAll](#)  
[FSWideStringArrayRemoveAt](#)

## Functions

### Functions summary

[FSWideStringArrayAdd](#)

Adds an element at the tail. Potentially growing the array

[FSWideStringArrayDestroy](#)

Destroys the wide-string array.

[FSWideStringArrayGetAt](#)

Retrieves an element specified by an index number.

[FSWideStringArrayGetSize](#)

Gets the number of elements in the array.

[FSWideStringArrayNew](#)

Creates a new empty wide-string array.

[FSWideStringArrayRemoveAll](#)

Cleans up the array.

[FSWideStringArrayRemoveAt](#)

Removes a number of elements at specified position.

### Functions detail

[FSWideStringArrayAdd](#)

**Syntax**

```
void FSWideStringArrayAdd (
    FS\_WideStringArray arr,
    FS\_LPCWSTR newItem
);
```

**Description**

Adds an element at the tail. Potentially growing the array

**Parameter**

---

arr	[In] The input wide-string array.
-----	-----------------------------------

---

newItem	[In] The input element.
---------	-------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1979

**FSWideStringArrayDestroy****Syntax**

```
void FSWideStringArrayDestroy (
    FS\_WideStringArray arr
);
```

**Description**

Destroys the wide-string array.

**Parameter**

---

arr	[In] The input wide-string array.
-----	-----------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1941

**FSWideStringArrayGetAt****Syntax**

```
void FSWideStringArrayGetAt (
    FS\_WideStringArray arr,
    FS\_INT32 index,
    FS\_WideString* outWideString
);
```

**Description**

Retrieves an element specified by an index number.

**Parameter**

arr	[In] The input wide-string array.
index	[In] Specifies the zero-based index of the element.
outWideString	[Out] Retrieves an element specified by an index number.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1950

**FSWideStringArrayGetSize****Syntax**

```
FS_INT32 FSWideStringArrayGetSize (
    FS\_WideStringArray arr
);
```

**Description**

Gets the number of elements in the array.

**Parameter**

arr	[In] The input wide-string array.
-----	-----------------------------------

**Return**

The number of elements in the array.

**Head file reference**

fs\_basicTempl.h: 1961

**FSWideStringArrayNew****Syntax**

```
FS_WideStringArray FSWideStringArrayNew (void );
```

**Description**

Creates a new empty wide-string array.

**Return**

A new empty wide-string array.

**Head file reference**

fs\_basicTempl.h: 1932

**FSWideStringArrayRemoveAll****Syntax**

```
void FSWideStringArrayRemoveAll (
    FS\_WideStringArray arr
);
```

**Description**

Cleans up the array.

**Parameter**

---

arr	[In] The input wide-string array.
-----	-----------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1970

**FSWideStringArrayRemoveAt****Syntax**

```
void FSWideStringArrayRemoveAt (
    FS\_WideStringArray arr,
    FS\_INT32 index
);
```

**Description**

Removes a number of elements at specified position.

**Parameter**

---

arr	[In] The input wide-string array.
-----	-----------------------------------

---

index	[In] Specifies the zero-based index in the array.
-------	---

---

**Return**

void

**Head file reference**  
fs\_basicTempl.h: 1989

## FS\_WordArray

### [Return from Used by](#)

### Description

A word array. See [FSWordArrayNew](#) , [FSWordArrayDestroy](#) .

### Returned from

### [FSWordArrayNew](#)

### Used by

[FRDocDidDeletePages](#)  
[FRDocWillDeletePages](#)  
[FSWordArrayAdd](#)  
[FSWordArrayAppend](#)  
[FSWordArrayCopy](#)  
[FSWordArrayDestroy](#)  
[FSWordArrayFind](#)  
[FSWordArrayGetAt](#)  
[FSWordArrayGetDataPtr](#)  
[FSWordArrayGetSize](#)  
[FSWordArrayGetUpperBound](#)  
[FSWordArrayInsertAt](#)  
[FSWordArrayInsertNewArrayAt](#)  
[FSWordArrayRemoveAll](#)  
[FSWordArrayRemoveAt](#)  
[FSWordArraySetAt](#)  
[FSWordArraySetAtGrow](#)  
[FSWordArraySetSize](#)

## Functions

### Functions summary

#### [FSWordArrayAdd](#)

Adds an element at the tail. Potentially growing the array

#### [FSWordArrayAppend](#)

Appends an array.

#### [FSWordArrayCopy](#)

Copies from an array.

#### [FSWordArrayDestroy](#)

Destroys a word array.

#### [FSWordArrayFind](#)

Finds an element from specified position to last

#### [FSWordArrayGetAt](#)

Retrieves an element specified by an index number.

**FSWordArrayGetDataPtr**

Gets a pointer to the specified element in the array. Direct pointer access.

**FSWordArrayGetSize**

Gets the number of elements in the array.

**FSWordArrayGetUpperBound**

Gets the upper bound in the array, actually the maximum valid index.

**FSWordArrayInsertAt**

Inserts one or more continuous element at specified position.

**FSWordArrayInsertNewArrayAt**

Inserts an array at specified position.

**FSWordArrayNew**

Creates a new empty word array.

**FSWordArrayRemoveAll**

Cleans up the array.

**FSWordArrayRemoveAt**

Removes a number of elements at specified position.

**FSWordArraySetAt**

Overwrites an element specified by an index number.

**FSWordArraySetAtGrow**

Sets an element value at specified position. Potentially growing the array.

**FSWordArraySetSize**

Changes the allocated size and the growing amount.

## Functions detail

### FSWordArrayAdd

#### Syntax

```
FS_INT32 FSWordArrayAdd (
    FS_WordArray arr,
    FS_WORD newItem
);
```

#### Description

Adds an element at the tail. Potentially growing the array

#### Parameter

---

arr	[In] The input word array.
-----	----------------------------

---

newItem	[In] The input element.
---------	-------------------------

---

#### Return

The added element's index.

#### Head file reference

fs\_basicTempl.h: 1562

## FSWordArrayAppend

### Syntax

```
FS_INT32 FSWordArrayAppend (
    FS\_WordArray arr,
    const FS\_WordArray srcArr
);
```

### Description

Appends an array.

### Parameter

---

arr	[In] The input word array.
-----	----------------------------

---

srcArr	[In] The input array.
--------	-----------------------

---

### Return

The old size in elements.

### Head file reference

fs\_basicTempl.h: 1572

## FSWordArrayCopy

### Syntax

```
void FSWordArrayCopy (
    FS\_WordArray arr,
    FS\_WordArray srcArr
);
```

### Description

Copies from an array.

### Parameter

---

arr	[In] The input word array.
-----	----------------------------

---

srcArr	[In] The input array.
--------	-----------------------

---

### Return

void

### Head file reference

fs\_basicTempl.h: 1582

## FSWordArrayDestroy

### Syntax

```
void FSWordArrayDestroy (
    FS\_WordArray arr
);
```

### Description

Destroys a word array.

### Parameter

---

arr	[In] The input word array.
-----	----------------------------

---

### Return

void

### Head file reference

[fs\\_basicTempl.h](#): 1483

## FSWordArrayFind

### Syntax

```
FS_INT32 FSWordArrayFind (
    FS\_WordArray arr,
    FS\_WORD item,
    FS\_INT32 nstartIndex
);
```

### Description

Finds an element from specified position to last

### Parameter

---

arr	[In] The input word array.
-----	----------------------------

---

item	[In] The input element.
------	-------------------------

---

nstartIndex	[In] Specifies the zero-based index of start element to find.
-------------	---

---

### Return

An index of the found element. It can be -1 for found none.

### Head file reference

[fs\\_basicTempl.h](#): 1636

## FSWordArrayGetAt

### Syntax

```
FS_WORD FSWordArrayGetAt (
    FS_WordArray arr,
    FS_INT32 index
);
```

### Description

Retrieves an element specified by an index number.

### Parameter

---

arr	[In] The input word array.
-----	----------------------------

---

index	[In] Specifies the zero-based index of the element.
-------	---

---

### Return

An element

### Head file reference

fs\_basicTempl.h: 1530

## FSWordArrayGetDataPtr

### Syntax

```
FS_WORD* FSWordArrayGetDataPtr (
    FS_WordArray arr,
    FS_INT32 index
);
```

### Description

Gets a pointer to the specified element in the array. Direct pointer access.

### Parameter

---

arr	[In] The input word array.
-----	----------------------------

---

index	[In] Specifies the zero-based index of element in the array.
-------	--

---

### Return

A pointer to the specified element.

### Head file reference

fs\_basicTempl.h: 1592

## FSWordArrayGetSize

### Syntax

```
FS_INT32 FSWordArrayGetSize (
    FS\_WordArray arr
);
```

### Description

Gets the number of elements in the array.

### Parameter

---

arr	[In] The input word array.
-----	----------------------------

---

### Return

The number of elements in the array.

### Head file reference

fs\_basicTempl.h: 1492

## FSWordArrayGetUpperBound

### Syntax

```
FS_INT32 FSWordArrayGetUpperBound (
    FS\_WordArray arr
);
```

### Description

Gets the upper bound in the array, actually the maximum valid index.

### Parameter

---

arr	[In] The input word array.
-----	----------------------------

---

### Return

The upper bound.

### Head file reference

fs\_basicTempl.h: 1501

## FSWordArrayInsertAt

### Syntax

```
void FSWordArrayInsertAt (
    FS\_WordArray arr,
    FS\_INT32 index,
    FS\_WORD newItem,
    FS\_INT32 nCount
```

);

**Description**

Inserts one or more continuous element at specified position.

**Parameter**

arr	[In] The input word array.
index	[In] Specifies the zero-based index in the array.
newItem	[In] Specifies the element value to insert.
nCount	[In] Specifies the count of the element to insert.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1602

**FSWordArrayInsertNewArrayAt****Syntax**

```
void FSWordArrayInsertNewArrayAt (
    FS_WordArray arr,
    FS_INT32 nstartIndex,
    FS_WordArray newArray
);
```

**Description**

Inserts an array at specified position.

**Parameter**

arr	[In] The input word array.
nstartIndex	[In] Specifies the zero-based index of start element to insert at.
newArray	[In] The input array.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1625

## FSWordArrayNew

### Syntax

```
FS_WordArray FSWordArrayNew (void );
```

### Description

Creates a new empty word array.

### Return

A new empty word array.

### Head file reference

fs\_basicTempl.h: 1474

## FSWordArrayRemoveAll

### Syntax

```
void FSWordArrayRemoveAll (
    FS\_WordArray arr
);
```

### Description

Cleans up the array.

### Parameter

---

arr	[In] The input word array.
-----	----------------------------

---

### Return

void

### Head file reference

fs\_basicTempl.h: 1521

## FSWordArrayRemoveAt

### Syntax

```
void FSWordArrayRemoveAt (
    FS\_WordArray arr,
    FS\_INT32 index,
    FS\_INT32 nCount
);
```

### Description

Removes a number of elements at specified position.

**Parameter**

---

arr	[In] The input word array.
-----	----------------------------

---

index	[In] Specifies the zero-based index in the array.
-------	---

---

nCount	[In] Specifies the count of element to remove.
--------	--

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1614

**FSWordArraySetAt****Syntax**

```
void FSWordArraySetAt (
    FS\_WordArray arr,
    FS\_INT32 index,
    FS\_WORD newItem
);
```

**Description**

Overwrites an element specified by an index number.

**Parameter**

---

arr	[In] The input word array.
-----	----------------------------

---

index	[In] Specifies the zero-based index of the element.
-------	---

---

newItem	[In] An element
---------	-----------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1540

**FSWordArraySetAtGrow**

**Syntax**

```
void FSWordArraySetAtGrow (
    FS\_WordArray arr,
    FS\_INT32 index,
    FS\_WORD newItem
);
```

**Description**

Sets an element value at specified position. Potentially growing the array.

**Parameter**

arr	[In] The input word array.
index	[In] Specifies the zero-based index of element in the array.
newItem	[In] The input element.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1551

**FSWordArraySetSize****Syntax**

```
void FSWordArraySetSize (
    FS\_WordArray arr,
    FS\_INT32 nNewSize,
    FS\_INT32 nGrowBy
);
```

**Description**

Changes the allocated size and the growing amount.

**Parameter**

arr	[In] The input word array.
nNewSize	[In] The new size in elements expected.

  

nGrowBy	[In] The growing amount in elements expected. <i>nGrowBy</i> can be -1 for the growing amount unchanged.
---------	--

**Return**

void

**Head file reference**

fs\_basicTempl.h: 1510

## FS\_XMLElement

[Return from Used by](#)

### Description

An object representing a XML element. You parse the XML data to get a XML element object, and then retrieve the attributes. You can also create an empty XML element object, and then edit the attributes, finally output the XML data. See [FSXMLElementparse](#) , [FSXMLElementNew](#) , [FSXMLElementSetTag](#) , [FSXMLElementOutputStream](#) .

### Returned from

[FSXMLElementClone](#)  
[FSXMLElementGetElement](#)  
[FSXMLElementGetElement2](#)  
[FSXMLElementGetElement3](#)  
[FSXMLElementGetParent](#)  
[FSXMLElementNew](#)  
[FSXMLElementNew2](#)  
[FSXMLElementNew3](#)  
[FSXMLElementparse](#)  
[FSXMLElementparse2](#)

### Used by

[FSXMLElementAddChildContent](#)  
[FSXMLElementAddChildElement](#)  
[FSXMLElementClone](#)  
[FSXMLElementCountAttrs](#)  
[FSXMLElementCountChildren](#)  
[FSXMLElementCountElements](#)  
[FSXMLElementDestroy](#)  
[FSXMLElementFindElement](#)  
[FSXMLElementGetAttrByIndex](#)  
[FSXMLElementGetAttrFloat](#)  
[FSXMLElementGetAttrFloat2](#)  
[FSXMLElementGetAttrInteger](#)  
[FSXMLElementGetAttrInteger2](#)  
[FSXMLElementGetAttrValue](#)  
[FSXMLElementGetAttrValue2](#)  
[FSXMLElementGetChildType](#)  
[FSXMLElementGetContent](#)  
[FSXMLElementGetElement](#)  
[FSXMLElementGetElement2](#)  
[FSXMLElementGetElement3](#)  
[FSXMLElementGetNamespace](#)  
[FSXMLElementGetNamespaceURI](#)

[FSXMLElementGetParent](#)  
[FSXMLElementGetTagName](#)  
[FSXMLElementHasAttr](#)  
[FSXMLElementInsertChildContent](#)  
[FSXMLElementInsertChildElement](#)  
[FSXMLElementOutputStream](#)  
[FSXMLElementOutputStream2](#)  
[FSXMLElementRemoveAttr](#)  
[FSXMLElementRemoveChild](#)  
[FSXMLElementRemoveChildren](#)  
[FSXMLElementSetAttrValue](#)  
[FSXMLElementSetAttrValue2](#)  
[FSXMLElementSetAttrValue3](#)  
[FSXMLElementSetTag](#)  
[FSXMLElementSetTag2](#)

## Enumerations

### Enumerations summary

#### [FS\\_Child\\_Type](#)

Children Type of FS\_XMLElement.

### Enumerations detail

#### FS\_Child\_Type

##### **Syntax**

```
enum FS_Child_Type{  
    Invalid,  
    Element,  
    Content  
};
```

##### **Description**

Children Type of FS\_XMLElement.

##### **Head file reference**

fs\_basicExpT.h: 1429

##### **Invalid**

Invalid.

##### **Element**

Element.

##### **Content**

Content.

## Functions

### Functions summary

**FSXMLElementAddChildContent**

Adds a content child.

**FSXMLElementAddChildElement**

Adds an element to the child list (at the last position).

**FSXMLElementClone**

Clones the specified XML element.

**FSXMLElementCountAttrs**

Gets the count of attributes of the input XML element.

**FSXMLElementCountChildren**

Gets the number of children of this element, including content segments and sub-elements.

**FSXMLElementCountElements**

Counts number of elements with particular tag.

**FSXMLElementDestroy**

Destroys the input XML element.

**FSXMLElementFindElement**

Finds an element and returns its index.

**FSXMLElementGetAttrByIndex**

Gets the attribute by index.

**FSXMLElementGetAttrFloat**

Gets a float from a particular attribute without a namespace.

**FSXMLElementGetAttrFloat2**

Gets a float from a particular attribute, using a namespace.

**FSXMLElementGetAttrInteger**

Gets an integer from a particular attribute without a namespace.

**FSXMLElementGetAttrInteger2**

Gets an integer from a particular attribute, using a namespace.

**FSXMLElementGetAttrValue**

Gets attribute value without a namespace.

**FSXMLElementGetAttrValue2**

Gets attribute value. Encoded in UTF-16LE format.

**FSXMLElementGetChildType**

Gets the type of a child, it can be either a content segment, or a sub-element.

**FSXMLElementGetContent**

Gets a content segment. If this child is an element, *outContent* will receive nothing.

**FSXMLElementGetElement**

Gets a particular child element. If this child is not an element, NULL will be returned.

**FSXMLElementGetElement2**

Gets an element with particular tag. If more than one element with the same tag, only the first one is returned.

**FSXMLElementGetElement3**

Gets an element with particular tag, and specified index if more than one elements with the same tag.

**FSXMLElementGetNamespace**

Gets the namespace of the XML element.

**FSXMLElementGetNamespaceURI**

Gets the full URI value for a qualified namespace.

**FSXMLElementGetParent**

Gets the parent element of the input XML element.

**FSXMLElementGetTagName**

Gets the tag name of the XML element.

**FSXMLElementHasAttr**

Determines whether a qualified attribute exists or not.

**FSXMLElementInsertChildContent**

Inserts a content segment before the specified index.

**FSXMLElementInsertChildElement**

Inserts a child element before the specified index

**FSXMLElementNew**

Creates an empty XML element, with a qualified namespace and tag name.

**FSXMLElementNew2**

Creates an empty element, with qualified tag name.

**FSXMLElementNew3**

Creates an empty element.

**FSXMLElementOutputStream**

Outputs to a XML stream.

**FSXMLElementOutputStream2**

Outputs through the file write handler.

**FSXMLElementparse**

Constructs an XML element using data in specified buffer. The buffer can be discarded immediately after construction finished, because at this time, all data have been loaded into the element (and its descendant element).

**FSXMLElementparse2**

Parses XML contents from a [FS\\_FileReadHandler](#) object. The [FS\\_FileReadHandler](#) object can be discarded immediately after construction finished, because at this time, all data have been loaded into the element (and its descendant element).

**FSXMLElementRemoveAttr**

Removes an attribute.

**FSXMLElementRemoveChild**

Removes a specified child item (element or content).

**FSXMLElementRemoveChildren**

Remove all children.

**FSXMLElementSetAttrValue**

Sets attribute with a wide string.

**FSXMLElementSetAttrValue2**

Sets attribute with an integer.

**FSXMLElementSetAttrValue3**

Sets attribute with a float.

**FSXMLElementSetTag**

Sets the tag name of the element.

**FSXMLElementSetTag2**

Sets the tag name of the element.

## Functions detail

### FSXMLElementAddChildContent

#### Syntax

```
void FSXMLElementAddChildContent (
    FS\_XMLElement XMLElement,
    FS\_WideString content,
    FS\_BOOL bCDATA
);
```

**Description**

Adds a content child.

**Parameter**


---

XMLElement	[In] The input XML element.
content	[In] The input content.
bCDATA	[In] Whether the content is CDATA or not. Sets it <a href="#">FALSE</a> as default.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3307

**Related method**

[FSXMLElementCountChildren](#)  
[FSXMLElementAddChildElement](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FSXMLElementAddChildElement****Syntax**

```
void FSXMLElementAddChildElement (
    FS\_XMLElement XMLElement,
    FS\_XMLElement pElement
);
```

**Description**

Adds an element to the child list (at the last position).

**Parameter**


---

XMLElement	[In] The input XML element.
pElement	[In] The input element.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3295

**Related method**

[FSXMLElementCountChildren](#)

**Note:**The child element can't be directly destroyed any more.

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FSXMLElementClone**

**Syntax**

```
FS_XMLElement FSXMLElementClone (
    FS_XMLElement XMLElement
);
```

**Description**

Clones the specified XML element.

**Parameter**

---

XMLElement	[In] The input XML element.
------------	-----------------------------

---

**Return**

The cloned XML element.

**Head file reference**

fs\_basicTempl.h: 3395

**Related method**

[FSXMLElementparse2](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 8.0.2](#)

**FSXMLElementCountAttrs**

**Syntax**

```
FS_DWORD FSXMLElementCountAttrs (
    FS_XMLElement XMLElement
);
```

**Description**

Gets the count of attributes of the input XML element.

**Parameter**

---

XMLElement	[In] The input XML element.
------------	-----------------------------

---

**Return**

The count of attributes of the input XML element.

**Head file reference**

fs\_basicTempl.h: 2992

**Related method**

[FSXMLElementGetAttrByIndex](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FSXMLElementCountChildren

**Syntax**

```
FS_DWORD FSXMLElementCountChildren (
    FS\_XMLElement XMLElement
);
```

**Description**

Gets the number of children of this element, including content segments and sub-elements.

**Parameter**

---

XMLElement	[In] The input XML element.
------------	-----------------------------

---

**Return**

The number of children.

**Head file reference**

fs\_basicTempl.h: 2987

**Related method**

[FSXMLElementGetChildType](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FSXMLElementCountElements

**Syntax**

```
FS_DWORD FSXMLElementCountElements (
    FS\_XMLElement XMLElement,
    FS\_LPCSTR space,
    FS\_LPCSTR tag
);
```

**Description**

Counts number of elements with particular tag.

**Parameter**

XMLElement	[In] The input XML element.
space	[In] The qualified namespace.
tag	[In] The local name for the tag.

**Return**

The count number of elements with the input tag.

**Head file reference**

fs\_basicTempl.h: 3190

**Related method**

[FSXMLElementGetElement](#)  
[FSXMLElementGetElement2](#)  
[FSXMLElementGetElement3](#)

**Since**

[SDK LATEST VERSION > 2.0](#)

**FSXMLElementDestroy****Syntax**

```
void FSXMLElementDestroy (
    FS\_XMLElement XMLElement
);
```

**Description**

Destroys the input XML element.

**Parameter**

XMLElement	[In] The input XML element.
------------	-----------------------------

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2903

**Related method**[FSXMLElementNew](#)[FSXMLElementNew2](#)[FSXMLElementNew3](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FSXMLElementFindElement****Syntax**

```
FS_DWORD FSXMLElementFindElement (
    FS\_XMLElement XMLElement,
    FS\_XMLElement pChild
);
```

**Description**

Finds an element and returns its index.

**Parameter**

---

XMLElement	[In] The input XML element.
------------	-----------------------------

---

pChild	[In] The input child element.
--------	-------------------------------

---

**Return**

The index number of child element, -1 if not found.

**Head file reference**

fs\_basicTempl.h: 3205

**Related method**[FSXMLElementCountElements](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FSXMLElementGetAttrByIndex****Syntax**

```
void FSXMLElementGetAttrByIndex (
    FS\_XMLElement XMLElement,
    FS\_INT32 index,
    FS\_ByteString* space,
    FS\_ByteString* name,
    FS\_WideString* value
);
```

**Description**

Gets the attribute by index.

**Parameter**

XMLElement	[In] The input XML element.
index	[In] Index of the attribute (start from 0).
space	[Out] It receives the qualified namespace of attribute name.
name	[Out] It receives the attribute name (local name).
value	[Out] It receive the attribute value.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2998

**Related method**

[FSXMLElementCountAttrs](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FSXMLElementGetAttrFloat****Syntax**

```
FS_BOOL FSXMLElementGetAttrFloat (
    FS_XMLElement XMLElement,
    FS_LPCSTR name,
    FS_FLOAT* attribute
);
```

**Description**

Gets a float from a particular attribute without a namespace.

**Parameter**

XMLElement	[In] The input XML element.
------------	-----------------------------

---

name	[In] The qualified attribute name.
------	------------------------------------

---

attribute	[Out] It receives the attribute value, a float value.
-----------	---

---

**Return**

Whether the attribute exists or not.

**Head file reference**

fs\_basicTempl.h: 3086

**Related method**

[FSXMLElementGetAttrFloat2](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FSXMLElementGetAttrFloat2****Syntax**

```
FS_BOOL FSXMLElementGetAttrFloat2 (
    FS_XMLElement XMLElement,
    FS_LPCSTR space,
    FS_LPCSTR name,
    FS_FLOAT* attribute
);
```

**Description**

Gets an float from a particular attribute, using a namespace.

**Parameter**


---

XMLElement	[In] The input XML element.
------------	-----------------------------

---

space	[In] The qualified namespace.
-------	-------------------------------

---

name	[In] The local name for the attribute.
------	--

---

attribute	[Out] It receives the attribute value, a float value.
-----------	---

---

**Return**

Whether the attribute exists or not.

**Head file reference**

fs\_basicTempl.h: 3094

**Related method**

[FSXMLElementGetAttrFloat](#)**Since**[SDK LATEEST VERSION > 2.0](#)[FSXMLElementGetAttrInteger](#)**Syntax**

```
FS_BOOL FSXMLElementGetAttrInteger (
    FS_XMLElement XMLElement,
    FS_LPCSTR name,
    FS_INT32* attribute
);
```

**Description**

Gets an integer from a particular attribute without a namespace.

**Parameter**

XMLElement	[In] The input XML element.
name	[In] The qualified attribute name.
attribute	[Out] It receives the attribute value, a integer value.

**Return**

Whether the attribute exists or not.

**Head file reference**

fs\_basicTempl.h: 3059

**Related method**[FSXMLElementGetAttrInteger2](#)**Since**[SDK LATEEST VERSION > 2.0](#)[FSXMLElementGetAttrInteger2](#)**Syntax**

```
FS_BOOL FSXMLElementGetAttrInteger2 (
    FS_XMLElement XMLElement,
    FS_LPCSTR space,
    FS_LPCSTR name,
    FS_INT32* attribute
);
```

**Description**

Gets an integer from a particular attribute, using a namespace.

**Parameter**


---

XMLElement	[In] The input XML element.
space	[In] The qualified namespace.
name	[In] The local name for the attribute.
attribute	[Out] It receives the attribute value, a integer value.

---

**Return**

Whether the attribute exists or not.

**Head file reference**

fs\_basicTempl.h: 3067

**Related method**

[FSXMLElementGetAttrInteger](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FSXMLElementGetAttrValue****Syntax**

```
FS_BOOL FSXMLElementGetAttrValue (
    FS_XMLElement XMLElement,
    FS_LPCSTR name,
    FS_WideString* attribute
);
```

**Description**

Gets attribute value without a namespace.

**Parameter**


---

XMLElement	[In] The input XML element.
name	[In] The qualified attribute name.
attribute	[Out] It receive the attribute value.

---

**Return**

Whether the attribute exists or not.

**Head file reference**

fs\_basicTempl.h: 3031

**Related method**

[FSXMLElementHasAttr](#)  
[FSXMLElementGetAttrValue2](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FSXMLElementGetAttrValue2**

**Syntax**

```
FS_BOOL FSXMLElementGetAttrValue2 (
    FS_XMLElement XMLElement,
    FS_LPCSTR space,
    FS_LPCSTR name,
    FS_WideString* attribute
);
```

**Description**

Gets attribute value. Encoded in UTF-16LE format.

**Parameter**

XMLElement	[In] The input XML element.
space	[In] The qualified namespace.
name	[In] The qualified attribute name.
attribute	[Out] It receive the attribute value, A UTF-16LE wide string.

**Return**

Whether the attribute exists or not.

**Head file reference**

fs\_basicTempl.h: 3040

**Related method**

[FSXMLElementGetAttrValue](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)**FSXMLElementGetChildType****Syntax**

```
FS_Child_Type FSXMLElementGetChildType (
    FS\_XMLElement XMLElement,
    FS\_DWORD index
);
```

**Description**

Gets the type of a child, it can be either a content segment, or a sub-element.

**Parameter**


---

<a href="#">XMLElement</a>	[In] The input XML element.
<a href="#">index</a>	[In] Specifies the zero-based index in children array.

---

**Return**

The type of specified child.

**Head file reference**

[fs\\_basicTempl.h](#): 3119

**Related method**

[FSXMLElementCountChildren](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FSXMLElementGetContent****Syntax**

```
void FSXMLElementGetContent (
    FS\_XMLElement XMLElement,
    FS\_DWORD index,
    FS\_WideString* outContent
);
```

**Description**

Gets a content segment. If this child is an element, *outContent* will receive nothing.

**Parameter**


---

<a href="#">XMLElement</a>	[In] The input XML element.
----------------------------	-----------------------------

---

---

index	[In] Specifies the zero-based index in children array.
-------	--

---

outContent	[Out] It receives the content.
------------	--------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3136

**Related method**[FSXMLElementGetChildType](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FSXMLElementGetElement****Syntax**

```
FS_XMLElement FSXMLElementGetElement (
    FS\_XMLElement XMLElement,
    FS\_DWORD index
);
```

**Description**

Gets a particular child element. If this child is not an element, NULL will be returned.

**Parameter**


---

XMLElement	[In] The input XML element.
------------	-----------------------------

---

index	[In] Specifies the zero-based index in children array.
-------	--

---

**Return**

A child element.

**Head file reference**

fs\_basicTempl.h: 3149

**Related method**[FSXMLElementGetChildType](#)[FSXMLElementGetElement2](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FSXMLElementGetElement2****Syntax**

```
FS_XMLEMENT FSXMLElementGetElement2 (
    FS_XMLEMENT XMLEMENT,
    FS_LPCSTR space,
    FS_LPCSTR tag
);
```

**Description**

Gets an element with particular tag. If more than one element with the same tag, only the first one is returned.

**Parameter**

XMLElement	[In] The input XML element.
space	[In] The qualified namespace.
tag	[In] The local name for the tag.

**Return**

An element.

**Head file reference**

fs\_basicTempl.h: 3157

**Related method**

[FSXMLElementGetElement](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FSXMLElementGetElement3****Syntax**

```
FS_XMLEMENT FSXMLElementGetElement3 (
    FS_XMLEMENT XMLEMENT,
    FS_LPCSTR space,
    FS_LPCSTR tag,
    FS_INT32 index
);
```

**Description**

Gets an element with particular tag, and specified index if more than one elements with the same tag.

**Parameter**

---

<code>XMLElement</code>	[In] The input XML element.
<code>space</code>	[In] The qualified namespace.
<code>tag</code>	[In] The input tag.
<code>index</code>	[In] Specifies the zero-based index of element with particular tag.

---

**Return**

An element.

**Head file reference**

`fs_basicTempl.h: 3175`

**Related method**

[FSXMLElementGetElement](#)  
[FSXMLElementGetElement2](#)

**Since**

[SDK LATEST VERSION > 2.0](#)

**FSXMLElementGetNamespace****Syntax**

```
void FSXMLElementGetNamespace (
    FS\_XMLElement XMLElement,
    FS\_BOOL bQualified,
    FS\_ByteString* outNamespace
);
```

**Description**

Gets the namespace of the XML element.

**Parameter**


---

<code>XMLElement</code>	[In] The input XML element.
<code>bQualified</code>	[In] Indicates whether return a qualified namespace or full URI namespace.
<code>outNamespace</code>	[Out] It receives the name space. If <i>bQualified</i> is TRUE, returns qualified namespace value, or full URI value.

---

**Return**

`void`

**Head file reference**

fs\_basicTempl.h: 2955

**Related method**[FSXMLElementNew](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FSXMLElementGetNamespaceURI****Syntax**

```
void FSXMLElementGetNamespaceURI (
    FS\_XMLElement XMLElement,
    FS\_LPCSTR name,
    FS\_ByteString* outNamespaceURI
);
```

**Description**

Gets the full URI value for a qualified namespace.

**Parameter**

---

XMLElement	[In] The input XML element.
------------	-----------------------------

---

name	[In] Qualified namespace.
------	---------------------------

---

outNamespaceURI	[Out] It receives the full URI value.
-----------------	---------------------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2968

**Related method**[FSXMLElementGetNamespace](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FSXMLElementGetParent****Syntax**

```
FS\_XMLElement FSXMLElementGetParent (
    FS\_XMLElement XMLElement
```

);

**Description**

Gets the parent element of the input XML element.

**Parameter**

---

XMLElement	[In] The input XML element.
------------	-----------------------------

---

**Return**

The parent XML element of the input XML element..

**Head file reference**

fs\_basicTempl.h: 2981

**Related method**

[FSXMLElementCountChildren](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FSXMLElementGetTagName

**Syntax**

```
void FSXMLElementGetTagName (
    FS\_XMLEMENT XMLEMENT,
    FS\_BOOL bQualified,
    FS\_BytString* outTagName
);
```

**Description**

Gets the tag name of the XML element.

**Parameter**

---

XMLElement	[In] The input XML element.
------------	-----------------------------

---

---

bQualified	[In] Indicates whether return a qualified tag name or not.
------------	--

---

---

outTagName	[Out] It receives the tag name. If <i>bQualified</i> is TRUE, the tag name is with qualified namespace, or only local tag name.
------------	---

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 2942

**Related method**[FSXMLElementNew](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FSXMLElementHasAttr****Syntax**

```
FS_BOOL FSXMLElementHasAttr (
    FS_XMLElement XMLElement,
    FS_LPCSTR qName
);
```

**Description**

Determines whether a qualified attribute exists or not.

**Parameter**

XMLElement	[In] The input XML element.
qName	[In] The input qualified attribute name.

**Return**

[TRUE](#) if attribute exists, [FALSE](#) if doesn't exist.

**Head file reference**

fs\_basicTempl.h: 3018

**Related method**[FSXMLElementCountAttrs](#)[FSXMLElementGetAttrByIndex](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FSXMLElementInsertChildContent****Syntax**

```
void FSXMLElementInsertChildContent (
    FS_XMLElement XMLElement,
    FS_DWORD index,
    FS_WideString content,
    FS_BOOL bCDATA
);
```

**Description**

Inserts a content segment before the specified index.

**Parameter**

XMLElement	[In] The input XML element.
index	[In] Specifies the zero-based index of element in the child array.
content	[In] The input content.
bCDATA	[In] Whether the content is CDATA or not. Sets it <a href="#">FALSE</a> as default.

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3334

**Related method**

[FSXMLElementAddChildContent](#)

**Since**

[SDK LATEST VERSION > 2.0](#)

**FSXMLElementInsertChildElement****Syntax**

```
void FSXMLElementInsertChildElement (
    FS\_XMLEMENT XMLElement,
    FS\_DWORD index,
    FS\_XMLEMENT pElement
);
```

**Description**

Inserts a child element before the specified index

**Parameter**

XMLElement	[In] The input XML element.
index	[In] Specifies the zero-based index of element in the child array.

pElement	[In] The input element.
----------	-------------------------

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3321

**Related method**

[FSXMLElementAddChildElement](#)

**Note:** The child element can't be directly destroyed any more.

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FSXMLElementNew****Syntax**

```
FS_XMLElement FSXMLElementNew (
    FS LPCSTR qSpace,
    FS LPCSTR tagName
);
```

**Description**

Creates an empty XML element, with a qualified namespace and tag name.

**Parameter**

---

qSpace	[In] Qualified namespace, or empty if no namespace or default namespace is used.
--------	--

---

tagName	[In] The input tag name.
---------	--------------------------

---

**Return**

An empty XML element, with a qualified namespace and tag name.

**Head file reference**

fs\_basicTempl.h: 2896

**Related method**

[FSXMLElementDestroy](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FSXMLElementNew2

### Syntax

```
FS_XMLElement FSXMLElementNew2 (
    FS_LPCSTR tagName
);
```

### Description

Creates an empty element, with qualified tag name.

### Parameter

---

tagName	[In] Qualified tag name.
---------	--------------------------

---

### Return

An empty element, with qualified tag name.

### Head file reference

fs\_basicTempl.h: 2908

### Related method

[FSXMLElementDestroy](#)

### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FSXMLElementNew3

### Syntax

```
FS_XMLElement FSXMLElementNew3 (void );
```

### Description

Creates an empty element.

### Return

An empty element.

### Head file reference

fs\_basicTempl.h: 2919

### Related method

[FSXMLElementDestroy](#)

### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FSXMLElementOutputStream

**Syntax**

```
void FSXMLElementOutputStream (
    FS\_XMLElement XMLElement,
    FS\_ByteString* output
);
```

**Description**

Outputs to a XML stream.

**Parameter**

XMLElement	[In] The input XML element.
output	[Out] It receives the XML stream.

**Return**

void

**Head file reference**

[fs\\_basicTempl.h: 3371](#)

**Related method**

[FSXMLElementOutputStream2](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FSXMLElementOutputStream2

**Syntax**

```
void FSXMLElementOutputStream2 (
    FS\_XMLElement XMLElement,
    FS\_FileWriteHandler handler
);
```

**Description**

Outputs through the file write handler.

**Parameter**

XMLElement	[In] The input XML element.
handler	[In] The input file write handler. Creates it by <a href="#">FSFileWriteHandlerNew</a> .

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3378

**Related method**[FSXMLElementOutputStream](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FSXMLElementparse****Syntax**

```
FS_XMLElement FSXMLElementparse (
    const void* pBuffer,
    FS\_DWORD size,
    FS\_BOOL bSaveSpaceChars,
    FS\_DWORD* pParsedSize
);
```

**Description**

Constructs an XML element using data in specified buffer. The buffer can be discarded immediately after construction finished, because at this time, all data have been loaded into the element (and its descendant element).

**Parameter**

pBuffer	[In] The input buffer.
size	[In] The size in bytes of the input buffer.
bSaveSpaceChars	[In] Indicates whether need save space characters in content string, TRUE if save, or FALSE. Sets it FALSE as default.
pParsedSize	[Out] It receives the parsed size. Sets it NULL as default if you don't want to receive it.

**Return**

The XML element.

**Head file reference**

fs\_basicTempl.h: 2865

**Related method**[FSXMLElementparse2](#)**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)**FSXMLElementparse2****Syntax**

```
FS_XMLElement FSXMLElementparse2 (
    FS_FileReadHandler fileReadHander,
    FS_BOOL bSaveSpaceChars,
    FS_DWORD* pParsedSize
);
```

**Description**

Parses XML contents from a [FS\\_FileReadHandler](#) object. The [FS\\_FileReadHandler](#) object can be discarded immediately after construction finished, because at this time, all data have been loaded into the element (and its descendant element).

**Parameter**

fileReadHander	[In] The input file access object. Creates it by <a href="#">FSFileReadHandlerNew</a> .
bSaveSpaceChars	[In] Indicates whether need save space characters in content string, TRUE if save, or FALSE. Sets it FALSE as default.
pParsedSize	[Out] It receives the parsed size. Sets it NULL as default if you don't want to receive it.

**Return**

The XML element.

**Head file reference**

fs\_basicTempl.h: 2876

**Related method**

[FSXMLElementparse](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FSXMLElementRemoveAttr****Syntax**

```
void FSXMLElementRemoveAttr (
    FS_XMLElement XMLElement,
    FS_LPCSTR name
);
```

**Description**

Removes an attribute.

**Parameter**

---

XMLElement	[In] The input XML element.
name	[In] The qualified attribute name.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3283

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FSXMLElementRemoveChild****Syntax**

```
void FSXMLElementRemoveChild (
    FS\_XMLElement XMLElement,
    FS\_DWORD index
);
```

**Description**

Removes a specified child item (element or content).

**Parameter**

---

XMLElement	[In] The input XML element.
index	[In] Specifies the zero-based index of element in the child array.

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3348

**Related method**

[FSXMLElementRemoveChildren](#)

**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FSXMLElementRemoveChildren****Syntax**

```
void FSXMLElementRemoveChildren (
    FS\_XMLElement XMLElement
);
```

**Description**

Remove all children.

**Parameter**

---

XMLElement	[In] The input XML element.
------------	-----------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3348

**Related method**[FSXMLElementRemoveChild](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FSXMLElementSetAttrValue****Syntax**

```
void FSXMLElementSetAttrValue (
    FS\_XMLElement XMLElement,
    FS\_LPCSTR name,
    FS\_WideString value
);
```

**Description**

Sets attribute with a wide string.

**Parameter**

---

XMLElement	[In] The input XML element.
------------	-----------------------------

---

name	[In] The qualified attribute name.
------	------------------------------------

---

---

value	[In] The input wide string.
-------	-----------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3243

**Related method**

[FSXMLElementGetAttrValue](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FSXMLElementSetAttrValue2****Syntax**

```
void FSXMLElementSetAttrValue2 (
    FS\_XMLElement XMLElement,
    FS\_LPCSTR name,
    FS\_INT32 value
);
```

**Description**

Sets attribute with an integer.

**Parameter**

---

XMLElement	[In] The input XML element.
------------	-----------------------------

---

---

name	[In] The qualified attribute name.
------	------------------------------------

---

---

value	[In] The input integer.
-------	-------------------------

---

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3256

**Related method**

[FSXMLElementGetAttrInteger](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

## FSXMLElementSetAttrValue3

### Syntax

```
void FSXMLElementSetAttrValue3 (
    FS_XMLElement XMLElement,
    FS_LPCSTR name,
    FS_FLOAT value
);
```

### Description

Sets attribute with a float.

### Parameter

XMLElement	[In] The input XML element.
name	[In] The qualified attribute name.
value	[In] The input float.

### Return

void

### Head file reference

fs\_basicTempl.h: 3269

### Related method

[FSXMLElementGetAttrFloat](#)

[FSXMLElementGetAttrFloat2](#)

### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FSXMLElementSetTag

### Syntax

```
void FSXMLElementSetTag (
    FS_XMLElement XMLElement,
    FS_LPCSTR qSpace,
    FS_LPCSTR tagname
);
```

### Description

Sets the tag name of the element.

### Parameter

---

<code>XMLElement</code>	[In] The input XML element.
-------------------------	-----------------------------

<code>qSpace</code>	[In] The qualified namespace for the tag.
---------------------	---

---

<code>tagname</code>	[In] The input tag name.
----------------------	--------------------------

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3217

**Related method**[FSXMLElementGetTagName](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FSXMLElementSetTag2****Syntax**

```
void FSXMLElementSetTag2 (
    FS\_XMLElement XMLElement,
    FS\_LPCSTR qTagName
);
```

**Description**

Sets the tag name of the element.

**Parameter**


---

<code>XMLElement</code>	[In] The input XML element.
-------------------------	-----------------------------

<code>qTagName</code>	[In] The input tag name.
-----------------------	--------------------------

**Return**

void

**Head file reference**

fs\_basicTempl.h: 3230

**Related method**[FSXMLElementGetTagName](#)[FSXMLElementSetTag](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)

# Reader Layer

---

## Object list

The Acrobat Support (AS) layer of the core API provides a variety of utility methods, including platform-independent memory allocation and fixed-point math utilities. In addition, it allows plug-ins to replace low-level file system routines used by Acrobat (including read, write, reopen, remove file, rename file, and other directory operations). This enables Acrobat to be used with other file systems, such as on-line systems. Several AS methods return error codes rather than raising exceptions on errors. This is because these methods are called at a low level where exception handling would be inconvenient and expensive.

## Object List

### General

#### FDRM\_CategoryRead

#### FDRM\_CategoryWrite

#### FDRM\_DescData

#### FDRM\_DescRead

#### FDRM\_DescWrite

#### FDRM\_EncryptDictRead

#### FDRM\_Encryptor

#### FDRM\_EnvelopeRead

#### FDRM\_FoacRead

#### FDRM\_FoacWrite

#### FDRM\_Mgr

#### FDRM\_PDFSchema

[\*\*FDRM\\_PKI\*\*](#)

[\*\*FDRM\\_PresentationData\*\*](#)

[\*\*FDRM\\_ScriptData\*\*](#)

[\*\*FDRM\\_SignatureData\*\*](#)

[\*\*FOFD\\_Action\*\*](#)

[\*\*FOFD\\_Actions\*\*](#)

[\*\*FOFD\\_CodeC\*\*](#)

[\*\*FOFD\\_Creator\*\*](#)

[\*\*FOFD\\_CryptoDict\*\*](#)

[\*\*FOFD\\_Dest\*\*](#)

[\*\*FOFD\\_Doc\*\*](#)

[\*\*FOFD\\_Outline\*\*](#)

[\*\*FOFD\\_Package\*\*](#)

[\*\*FOFD\\_Page\*\*](#)

[\*\*FOFD\\_Parser\*\*](#)

[\*\*FOFD\\_Perms\*\*](#)

[\*\*FOFD\\_PrintSetting\*\*](#)

[\*\*FOFD\\_Sign\*\*](#)

[\*\*FOFD\\_SM4CryptoHandler\*\*](#)

[\*\*FOFD\\_SMSecurityHandler\*\*](#)

**FOFD\_StdCertSecurityHandler**

**FOFD\_StdCryptoHandler**

**FOFD\_StdSecurityHandler**

**FOFD\_Sys**

**FOFD\_UIMgr**

**FOFD\_VPrefers**

**FOFD\_WriteDoc**

**FR\_ActionWizardData**

**FR\_ActionWizardHandler**

**FR\_Annot**

The UI layer annotation object.

**FR\_App**

FR\_App represents the Foxit Reader application itself.

**FR\_BulbMsgCenter**

**FR\_CloudLoginProvider**

**FR\_Cursor**

A data structure representing the cursor.

**FR\_CustomSignature**

**FR\_Document**

A FR\_Document is a view of a PDF document in a window.

**FR\_DocView**

A [FR\\_DocView](#) is the area of the Foxit Reader window that displays the contents of a document page. Every [FR\\_DocView](#) may have more than one [FR\\_PageView](#). It contains references to the [FPD Document](#) or the document being displayed.

## [FR Edit](#)

### [FR Edit FontMap](#)

### [FR Edit Iterator](#)

### [FR Edit UndoItem](#)

## [FR EmptyFrameWndViewEventHandler](#)

The [FR\\_EmptyFrameWndViewEventHandler](#) object represents the event handler of the empty frame window view. The plug-in can register the event handler to process the event when the plug-in creates the empty frame window. See [FRApCreateEmptyFrameViewEventHandler](#) and [FRApCreateAnEmptyFrameWnd2](#).

## [FR FormatTools](#)

The format tool is used to set the format of the PDF object and text. You can set the format like font name, font size, color and so on.

## [FR FuncBtn](#)

The plug-in can add a button to the left navigation panel bar.

## [FR HTMLMgr](#)

The [FR\\_HTMLMgr](#) object is used to manage the HTML windows in *Foxit Reader*. See [FRHTMLMgrGet](#).

## [FR Language](#)

The [FR\\_Language](#) object is used to change the language for Foxit Reader plug-in UI. You can create it by [FRLanguageCreate](#).

## [FR Menu](#)

A [FR\\_Menu](#) is a menu in the Foxit Reader's menu bar. Plug-ins can create new menus, add menu items at any location in any menu, and remove menu items. Deleting an [FR\\_Menu](#) removes it from the menu bar (if it was attached) and deletes all the menu items it contains. A [FR\\_Menu](#) have a name when it is added to menu bar, plug-ins can access a menu by using [FRMenuBarGetMenuByName](#) () or [FRMenuBarGetMenuByIndex](#) (). Submenus (also called pullright menus) are [FR\\_Menu](#) objects that are attached to an [FR\\_MenuItem](#) instead of the menu bar. Your plug-in cannot directly remove a submenu. Instead, it must remove the [FR\\_MenuItem](#) to which the submenu is attached.

## **FRMenuBar**

The [FRMenuBar](#) is the Foxit Reader menu bar and contains a list of all menus.

## **FRMenuItem**

A [FRMenuItem](#) is a menu item in a menu. It has attributes, including the following:

- A name that indicates the menu item.
- A title that displayed on screen.
- A procedure to execute when the menu item is selected.
- A procedure to compute whether the menu item is enabled.
- A procedure to compute whether the menu item is checkmarked, and whether it has a sub-menu.

## **FRMenuOwnerDrawHandler**

FR MenuOwnerDraw Handler.

## **FRMessageBar**

The [FRMessageBar](#) object is used to show some customer message. Plug-in can add some element to the ssage bar, such as bitmap, text, button and so on.

## **FPPageView**

A [FPPageView](#) is a view of a [FPDPage](#) object. The page view is a part(always a rectangle area) of document view.

## **FRPanelMgr**

The [FRPanelMgr](#) object is used to manage the navigation panels. You can get an existing panel manager through [FRPanelMgrGetPanelMgrFromChildFrm](#) . And can also create a new one through [FRPanelMgrNewPanelMgr](#) .

## **FRProfStore**

## **FRPropertyTools**

The property tools is used to set line color and opacity of the PDF object in classic mode.

## **FRResourcePropertyBox**

The property box is used to edit the properties of objects, such as the annotations, the pages and so on. The Foxit Reader creates the property box so that many types of objects can reuse the same property box to edit the properties.

## **FRResourcePropertyPage**

The property page of the property box.

## **FR\_ResourcePropertySource**

The property source of the property box.

## **FR\_RibbonBackStageViewItem**

The [FR\\_RibbonBackStageViewItem](#) object can be added under FILE category. The back stage view item is associated with a view. You can add your own dialog or property sheet page on the view. See [FRRibbonBackStageViewItemAddDialog](#) d [FRRibbonBackStageViewItemAddPropertySheetPage](#) .

## **FR\_RibbonBar**

The Foxit Reader has a ribbon bar in ribbon mode. The ribbon contains several categories. You can get the ribbon bar object by calling [FRAppGetRibbonBar](#) .

## **FR\_RibbonButton**

The ribbon button is a type of ribbon element.

## **FR\_RibbonCategory**

A [FR\\_RibbonCategory](#) object can be used to manage the operation categories. For example, all the commenting tools re included in the COMMENT category. For further classification, you have to add ribbon panels to the ribbon categories.

## **FR\_RibbonCheckBox**

The ribbon check box is a type of ribbon element.

## **FR\_RibbonColorButton**

The ribbon color button is a type of ribbon element.

## **FR\_RibbonComboBox**

The ribbon combo box is a type of ribbon element.

## **FR\_RibbonEdit**

The ribbon edit is a type of ribbon element.

## **FR\_RibbonElement**

The ribbon element object is a basic object. It manipulates the common properties of all types of ribbon element.

## **FR\_RibbonFontComboBox**

The ribbon font combo box is a type of ribbon element.

## **FR\_RibbonLabel**

The ribbon label is a type of ribbon element.

## **FR\_RibbonListButton**

The ribbon list button is a type of ribbon element.

## **FR\_RibbonPaletteButton**

The ribbon palette button is a type of ribbon element.

## **FR\_RibbonPanel**

The ribbon panel can be used to classify the operation. You can put the ribbon element on the ribbon panel. The ribbon panel object s managed by the ribbon category. You can add a ribbon panel through [FRRibbonCategoryAddPanel](#) .

## **FR\_RibbonRadioButton**

The ribbon radio button is a type of ribbon element.

## **FR\_RibbonSlider**

The ribbon slider is a type of ribbon element.

## **FR\_RibbonStyleButton**

The ribbon style button. You can modify the MFC button to the ribbon style button.

## **FR\_RibbonStyleListBox**

The ribbon style list box. You can modify the MFC list box to the ribbon style list box.

## **FR\_RibbonStyleStatic**

The ribbon style static box. You can modify the MFC static box to the ribbon style static box.

## **FR\_ScrollBarThumbnailView**

## **FR\_SpellCheck**

The [FR\\_SpellCheck](#) object can be used to check the spelling. See [FRSpellCheckNew](#).

## **FR\_StatusBar**

### **FR\_Sys**

It is used to manage the APIs of system-dependent. See [FRSysLoadStandarCursor](#), [FRSysGetCursor](#), [FRSysSetCursor](#).

## **FR\_TabBand**

A [FR\\_TabBand](#) is a band where documents' tabs are shown. A tab is associated with a window.

## **FR\_TextSelectTool**

The text select tool is used to process the operation of text selecting in the doc view.

## **FR\_ThumbnailView**

A [FR\\_ThumbnailView](#) is a view thumbnail view related to the document view. It displays the thumbnail of the document in the page panel.

## **FR\_Tool**

A [FR\\_Tool](#) is an object that can handle key presses and mouse events in the region of a document view.

## **FR\_ToolBar**

[FR\\_ToolBar](#) is the Foxit Reader toolbar. A plug-in can also create fly-out toolbars that contain dditional buttons and attach the fly-out toolbars to existing button.

## **FR\_ToolButton**

An [FR\\_ToolButton](#) is a button in the Foxit Reader's toolbar. Like menu items, the procedure that executes when the button is clicked can be set by a plug-in. Although not required, there generally is a menu item corresponding to each button, llowing users to select a function using either the button or the menu item.

## **FR\_Touchup**

## **FR\_UIProgress**

The [FR\\_UIProgress](#) object is referred to a progress bar. See [FRUIProgressCreate](#).

## **FR\_VariableText**

[\*\*FR\\_VariableText\\_Iterator\*\*](#)

[\*\*FR\\_VariableText\\_Provider\*\*](#)

[\*\*FR\\_VTLine\*\*](#)

[\*\*FR\\_VTSecProps\*\*](#)

[\*\*FR\\_VTSection\*\*](#)

[\*\*FR\\_VTWord\*\*](#)

[\*\*FR\\_VTWordPlace\*\*](#)

[\*\*FR\\_VTWordProps\*\*](#)

[\*\*FR\\_VTWordRange\*\*](#)

[\*\*FR\\_WindowsDIB\*\*](#)

[\*\*FRMS\\_Common\*\*](#)

[\*\*FRMS\\_Decryption\*\*](#)

[\*\*FRMS\\_Encryption\*\*](#)

## General

Description

Definitions

Definitions summary

[\*\*FR\\_CT\\_ANNOTATION\*\*](#)

The capture type of *Annotation* .

[\*\*FR\\_CT\\_TOUCHUP\*\*](#)

The capture type of *Touchup* .

[\*\*FR\\_PERM\\_ANNOTATE\*\*](#)

adding or modifying annotations, filling in forms

[\*\*FR\\_PERM\\_ASSEMBLE\*\*](#)

Assembling the document (page organizing)

[\*\*FR\\_PERM\\_EXTRACT\\_ACCESS\*\*](#)

extracting text or image for accessibility

**FR\_PERM\_EXTRACT\_COPY**

extracting text or image for copying

**FR\_PERM\_FILL\_FORM**

Filling in existing form

**FR\_PERM MODIFY\_CONTENT**

modifying page contents or form fields

**FR\_PERM\_PRINT**

printing

**FR\_PERM\_PRINT\_HIGN**

printing in high quality

**FRCIPHER\_AES**

FRCIPHER\_AES

**FRCIPHER\_NONE**

FRCIPHER\_NONE

**FRCIPHER\_RC4**

FRCIPHER\_RC4

**ACTION\_COPY**

The action name of Copy for data collection.

**ACTION\_CUT**

The action name of Cut for data collection.

**ACTION\_EDIT**

The action name of Edit for data collection.

**ACTION\_FIND**

The action name of Find for data collection.

**ACTION\_SELECT**

The action name of Select for data collection.

**FR\_FUNCTION\_CALLOUT**

The function name of Callout for data collection.

**FR\_FUNCTION\_FIND**

The function name of Find for data collection.

**FR\_FUNCTION\_HIGHLIGHT**

The function name of Highlight for data collection.

**FR\_FUNCTION\_INSERTTEXT**

The function name of InsertText for data collection.

**FR\_FUNCTION\_NOTE**

The function name of Note for data collection.

**FR\_FUNCTION\_REPLACETEXT**

The function name of ReplaceText for data collection.

**FR\_FUNCTION\_SELECTTEXT**

The function name of SelectText for data collection.

**FR\_FUNCTION\_SNAPSHOT**

The function name of SnapShot for data collection.

**FR\_FUNCTION\_SQUIGGLYUNDERLINE**

The function name of SquigglyUnderline for data collection.

**FR\_FUNCTION\_STRIKEOUT**

The function name of Strikeout for data collection.

**FR\_FUNCTION\_TEXTBOX**

The function name of Textbox for data collection.

**FR\_FUNCTION\_TYPEWRITER**

The function name of Typewriter for data collection.

**FR FUNCTION UNDERLINE**

The function name of Underline for data collection.

**FR OEM ASUS**

The OEM version of Asus.

**FR OEM GENERAL**

The general version for OEM.

**FR OEM HPCM**

The OEM version of HP commercial.

**FR OEM HPCS**

The OEM version of HP consumer.

**FR OEM LENOVO**

The OEM version of HP Lenovo.

**FR SIG SHOW ALL**

Shows all flags.

**FR SIG SHOW DATE**

Shows date on description.

**FR SIG SHOW DN**

Shows dn on description.

**FR SIG SHOW FOXITFLOGO**

Shows foxit logo on signature ap.

**FR SIG SHOW LABEL**

Shows label on description.

**FR SIG SHOW LOCATION**

Shows location on description.

**FR SIG SHOW NAME**

Shows signer name on description.

**FR SIG SHOW REASON**

Shows reason on description.

**FR SIG TIMESTAMP DOC**

It is the type of document timestamp.

**FR SIG TIMESTAMP EXPIRE**

The time stamp is expired.

**FR SIG TIMESTAMP INVALID**

The time stamp is invalid.

**FR SIG TIMESTAMP ISSUER ISUNKNOWN**

The issuer of the time stamp is unknown.

**FR SIG TIMESTAMP ISSUER ISVALID**

The issuer of the time stamp is valid.

**FR SIG TIMESTAMP NONE**

No timestamp.

**FR SIG TIMESTAMP VALID**

The time stamp is valid.

**FR SIG VERIFY CHANGE**

The document is changed beyond the limits.

**FR SIG VERIFY ERRORBYTERANGE**

The signature data bytes is not in the expected range.

**FR SIG VERIFY ERRORDATA**

The signature data is corrupted and can not be parsed successfully.

**FR SIG VERIFY INCREDIBLE**

The signature is not trusted.

**FR\_SIG\_VERIFY\_INVALID**

The signature state is invalid.

**FR\_SIG\_VERIFY\_ISSUER\_CURRENT**

The issuer is the current issuer.

**FR\_SIG\_VERIFY\_ISSUER\_EXPIRE**

The certificate is expired.

**FR\_SIG\_VERIFY\_ISSUER\_REVOKED**

The certificate is revoked.

**FR\_SIG\_VERIFY\_ISSUER\_UNCHECK**

The issuer is unchecked.

**FR\_SIG\_VERIFY\_ISSUER\_UNKNOW**

The issuer is unknown.

**FR\_SIG\_VERIFY\_ISSUER\_VALID**

The issuer is valid.

**FR\_SIG\_VERIFY\_NOSUPPORTWAY**

The signature type is not supported.

**FR\_SIG\_VERIFY\_VALID**

The signature state is valid.

**FR\_ST\_ANNOTATION**

The selection type of *Annotation*

**FR\_ST\_BITMAP**

The selection type of *Bitmap*

**FR\_ST\_BOOKMARK**

The selection type of *Bookmark*

**FR\_ST\_TEXT**

The selection type of *Text*

**FR\_ST\_THUMBNAIL**

The selection type of *Thumbnail*

**FR\_TASKPANE\_ADVSEARCH**

The name of advanced search task pane.

## Definitions detail

### FR\_CT\_ANNOTATION

#### Syntax

```
#define FR_CT_ANNOTATION "Annotation"
```

#### Description

The capture type of *Annotation*.

#### Group

[CaptureType](#)

#### Head file reference

fr\_appExpT.h: 4036

### FR\_CT\_TOUCHUP

**Syntax**

```
#define FR_CT_TOUCHUP "Touchup"
```

**Description**

The capture type of *Touchup* .

**Group**

[CaptureType](#)

**Head file reference**

fr\_appExpT.h: 4038

## FR\_PERM\_ANNOTATE

**Syntax**

```
#define FR_PERM_ANNOTATE 0x0020
```

**Description**

adding or modifying annotations, filling in forms

**Group**

[DocPermission](#)

**Head file reference**

fr\_appExpT.h: 2337

## FR\_PERM\_ASSEMBLE

**Syntax**

```
#define FR_PERM_ASSEMBLE 0x0400
```

**Description**

Assembling the document (page organizing)

**Group**

[DocPermission](#)

**Head file reference**

fr\_appExpT.h: 2343

## FR\_PERM\_EXTRACT\_ACCESS

**Syntax**

```
#define FR_PERM_EXTRACT_ACCESS 0x0200
```

**Description**

extracting text or image for accessibility

**Group**

[DocPermission](#)

**Head file reference**

fr\_appExpT.h: 2334

## FR\_PERM\_EXTRACT\_COPY

**Syntax**

```
#define FR_PERM_EXTRACT_COPY 0x0010
```

**Description**

extracting text or image for copying

**Group**

[DocPermission](#)

**Head file reference**

fr\_appExpT.h: 2331

## FR\_PERM\_FILL\_FORM

**Syntax**

```
#define FR_PERM_FILL_FORM 0x0100
```

**Description**

Filling in existing form

**Group**

[DocPermission](#)

**Head file reference**

fr\_appExpT.h: 2340

## FR\_PERM MODIFY\_CONTENT

**Syntax**

```
#define FR_PERM MODIFY_CONTENT 0x0008
```

**Description**

modifying page contents or form fields

**Group**

[DocPermission](#)

**Head file reference**

fr\_appExpT.h: 2328

## FR\_PERM\_PRINT

### Syntax

#define FR\_PERM\_PRINT 0x0004

### Description

printing

### Group

[DocPermission](#)

### Head file reference

fr\_appExpT.h: 2325

## FR\_PERM\_PRINT\_HIGN

### Syntax

#define FR\_PERM\_PRINT\_HIGN 0x0800

### Description

printing in high quality

### Group

[DocPermission](#)

### Head file reference

fr\_appExpT.h: 2346

## FRCIPHER\_AES

### Syntax

#define FRCIPHER\_AES 2

### Description

FRCIPHER\_AES

### Group

[FRCIPHERFlag](#)

### Head file reference

fr\_appExpT.h: 2362

## FRCIPHER\_NONE

**Syntax**

```
#define FRCIPHER_NONE 0
```

**Description**

FRCIPHER\_NONE

**Group**

[FRCIPHERFlag](#)

**Head file reference**

fr\_appExpT.h: 2358

## FRCIPHER\_RC4

**Syntax**

```
#define FRCIPHER_RC4 1
```

**Description**

FRCIPHER\_RC4

**Group**

[FRCIPHERFlag](#)

**Head file reference**

fr\_appExpT.h: 2360

## ACTION\_COPY

**Syntax**

```
#define ACTION_COPY L"Copy"
```

**Description**

The action name of Copy for data collection.

**Group**

[FRDataCollectionActionNames](#)

**Head file reference**

fr\_appExpT.h: 5423

## ACTION\_CUT

**Syntax**

```
#define ACTION_CUT L"Cut"
```

**Description**

The action name of Cut for data collection.

**Group**

[FRDataCollectionActionNames](#)

**Head file reference**

fr\_appExpT.h: 5425

## ACTION\_EDIT

**Syntax**

#define ACTION\_EDIT L"Edit"

**Description**

The action name of Edit for data collection.

**Group**

[FRDataCollectionActionNames](#)

**Head file reference**

fr\_appExpT.h: 5431

## ACTION\_FIND

**Syntax**

#define ACTION\_FIND L"Find"

**Description**

The action name of Find for data collection.

**Group**

[FRDataCollectionActionNames](#)

**Head file reference**

fr\_appExpT.h: 5427

## ACTION\_SELECT

**Syntax**

#define ACTION\_SELECT L"Select"

**Description**

The action name of Select for data collection.

**Group**

[FRDataCollectionActionNames](#)

**Head file reference**

fr\_appExpT.h: 5429

## FR\_FUNCTION\_CALLOUT

### Syntax

```
#define FR_FUNCTION_CALLOUT L"Callout"
```

### Description

The function name of Callout for data collection.

### Group

[FRDataCollectionFunctionNames](#)

### Head file reference

fr\_appExpT.h: 5406

## FR\_FUNCTION\_FIND

### Syntax

```
#define FR_FUNCTION_FIND L"Find"
```

### Description

The function name of Find for data collection.

### Group

[FRDataCollectionFunctionNames](#)

### Head file reference

fr\_appExpT.h: 5390

## FR\_FUNCTION\_HIGHLIGHT

### Syntax

```
#define FR_FUNCTION_HIGHLIGHT L"Highlight"
```

### Description

The function name of Highlight for data collection.

### Group

[FRDataCollectionFunctionNames](#)

### Head file reference

fr\_appExpT.h: 5392

## FR\_FUNCTION\_INSERTTEXT

**Syntax**

```
#define FR_FUNCTION_INSERTTEXT L"InsertText"
```

**Description**

The function name of InsertText for data collection.

**Group**

[FRDataCollectionFunctionNames](#)

**Head file reference**

fr\_appExpT.h: 5404

## FR\_FUNCTION\_NOTE

**Syntax**

```
#define FR_FUNCTION_NOTE L"Note"
```

**Description**

The function name of Note for data collection.

**Group**

[FRDataCollectionFunctionNames](#)

**Head file reference**

fr\_appExpT.h: 5410

## FR\_FUNCTION\_REPLACETEXT

**Syntax**

```
#define FR_FUNCTION_REPLACETEXT L"ReplaceText"
```

**Description**

The function name of ReplaceText for data collection.

**Group**

[FRDataCollectionFunctionNames](#)

**Head file reference**

fr\_appExpT.h: 5402

## FR\_FUNCTION\_SELECTTEXT

**Syntax**

```
#define FR_FUNCTION_SELECTTEXT L"SelectText"
```

**Description**

The function name of SelectText for data collection.

**Group**

[FRDataCollectionFunctionNames](#)

**Head file reference**

fr\_appExpT.h: 5388

## FR\_FUNCTION\_SNAPSHOT

**Syntax**

#define FR\_FUNCTION\_SNAPSHOT L"SnapShot"

**Description**

The function name of SnapShot for data collection.

**Group**

[FRDataCollectionFunctionNames](#)

**Head file reference**

fr\_appExpT.h: 5412

## FR\_FUNCTION\_SQUIGGLYUNDERLINE

**Syntax**

#define FR\_FUNCTION\_SQUIGGLYUNDERLINE L"SquigglyUnderline"

**Description**

The function name of SquigglyUnderline for data collection.

**Group**

[FRDataCollectionFunctionNames](#)

**Head file reference**

fr\_appExpT.h: 5394

## FR\_FUNCTION\_STRIKEOUT

**Syntax**

#define FR\_FUNCTION\_STRIKEOUT L"Strikeout"

**Description**

The function name of Strikeout for data collection.

**Group**

[FRDataCollectionFunctionNames](#)

**Head file reference**

fr\_appExpT.h: 5400

## FR\_FUNCTION\_TEXTBOX

### Syntax

```
#define FR_FUNCTION_TEXTBOX L"Textbox"
```

### Description

The function name of Textbox for data collection.

### Group

[FRDataCollectionFunctionNames](#)

### Head file reference

fr\_appExpT.h: 5408

## FR\_FUNCTION\_TYPEWRITER

### Syntax

```
#define FR_FUNCTION_TYPEWRITER L"Typewriter"
```

### Description

The function name of Typewriter for data collection.

### Group

[FRDataCollectionFunctionNames](#)

### Head file reference

fr\_appExpT.h: 5398

## FR\_FUNCTION\_UNDERLINE

### Syntax

```
#define FR_FUNCTION_UNDERLINE L"Underline"
```

### Description

The function name of Underline for data collection.

### Group

[FRDataCollectionFunctionNames](#)

### Head file reference

fr\_appExpT.h: 5396

## FR\_OEM\_ASUS

**Syntax**

```
#define FR_OEM_ASUS "Asus"
```

**Description**

The OEM version of Asus.

**Group**

[FROEMVersion](#)

**Head file reference**

fr\_appExpT.h: 5531

## FR\_OEM\_GENERAL

**Syntax**

```
#define FR_OEM_GENERAL "OEM"
```

**Description**

The general version for OEM.

**Group**

[FROEMVersion](#)

**Head file reference**

fr\_appExpT.h: 5525

## FR\_OEM\_HPCM

**Syntax**

```
#define FR_OEM_HPCM "HPCommercial"
```

**Description**

The OEM version of HP commercial.

**Group**

[FROEMVersion](#)

**Head file reference**

fr\_appExpT.h: 5527

## FR\_OEM\_HPCS

**Syntax**

```
#define FR_OEM_HPCS "HPConsumer"
```

**Description**

The OEM version of HP consumer.

**Group**[FROEMVersion](#)**Head file reference**

fr\_appExpT.h: 5529

**FR\_OEM\_LENOVO****Syntax**

#define FR\_OEM\_LENOVO "Lenovo"

**Description**

The OEM version of HP Lenovo.

**Group**[FROEMVersion](#)**Head file reference**

fr\_appExpT.h: 5533

**FR\_SIG\_SHOW\_ALL****Syntax**#define FR\_SIG\_SHOW\_ALL FR\_SIG\_SHOW\_NAME |  
FR\_SIG\_SHOW\_LOCATION | FR\_SIG\_SHOW\_DN | FR\_SIG\_SHOW\_DATE |  
FR\_SIG\_SHOW\_REASON | FR\_SIG\_SHOW\_LABEL |  
FR\_SIG\_SHOW\_FOXITFLOGO**Description**

Shows all flags.

**Group**[FRSIGShowAPFlags](#)**Head file reference**

fr\_docExpT.h: 217

**FR\_SIG\_SHOW\_DATE****Syntax**

#define FR\_SIG\_SHOW\_DATE 0x008

**Description**

Shows date on description.

**Group**[FRSIGShowAPFlags](#)

**Head file reference**

fr\_docExpT.h: 209

## FR\_SIG\_SHOW\_DN

**Syntax**

```
#define FR_SIG_SHOW_DN 0x004
```

**Description**

Shows dn on description.

**Group**

[FRSIGShowAPFlags](#)

**Head file reference**

fr\_docExpT.h: 207

## FR\_SIG\_SHOW\_FOXITFLOGO

**Syntax**

```
#define FR_SIG_SHOW_FOXITFLOGO 0x040
```

**Description**

Shows foxit logo on signature ap.

**Group**

[FRSIGShowAPFlags](#)

**Head file reference**

fr\_docExpT.h: 215

## FR\_SIG\_SHOW\_LABEL

**Syntax**

```
#define FR_SIG_SHOW_LABEL 0x020
```

**Description**

Shows label on description.

**Group**

[FRSIGShowAPFlags](#)

**Head file reference**

fr\_docExpT.h: 213

## FR\_SIG\_SHOW\_LOCATION

### Syntax

```
#define FR_SIG_SHOW_LOCATION 0x002
```

### Description

Shows location on description.

### Group

[FRSIGShowAPFlags](#)

### Head file reference

fr\_docExpT.h: 205

## FR\_SIG\_SHOW\_NAME

### Syntax

```
#define FR_SIG_SHOW_NAME 0x001
```

### Description

Shows signer name on description.

### Group

[FRSIGShowAPFlags](#)

### Head file reference

fr\_docExpT.h: 203

## FR\_SIG\_SHOW\_REASON

### Syntax

```
#define FR_SIG_SHOW_REASON 0x010
```

### Description

Shows reason on description.

### Group

[FRSIGShowAPFlags](#)

### Head file reference

fr\_docExpT.h: 211

## FR\_SIG\_TIMESTAMP\_DOC

### Syntax

```
#define FR_SIG_TIMESTAMP_DOC 0x00800000
```

**Description**

It is the type of document timestamp.

**Group**

[FRSIGVerifySignatureStates](#)

**Head file reference**

fr\_docExpT.h: 257

## FR\_SIG\_TIMESTAMP\_EXPIRE

**Syntax**

```
#define FR_SIG_TIMESTAMP_EXPIRE 0x04000000
```

**Description**

The time stamp is expired.

**Group**

[FRSIGVerifySignatureStates](#)

**Head file reference**

fr\_docExpT.h: 263

## FR\_SIG\_TIMESTAMP\_INVALID

**Syntax**

```
#define FR_SIG_TIMESTAMP_INVALID 0x02000000
```

**Description**

The time stamp is invalid.

**Group**

[FRSIGVerifySignatureStates](#)

**Head file reference**

fr\_docExpT.h: 261

## FR\_SIG\_TIMESTAMP\_ISSUER\_ISUNKNOWN

**Syntax**

```
#define FR_SIG_TIMESTAMP_ISSUER_ISUNKNOWN 0x08000000
```

**Description**

The issuer of the time stamp is unknown.

**Group**

[FRSIGVerifySignatureStates](#)**Head file reference**

fr\_docExpT.h: 265

**FR\_SIG\_TIMESTAMP\_ISSUER\_ISVALID****Syntax**

#define FR\_SIG\_TIMESTAMP\_ISSUER\_ISVALID 0x10000000

**Description**

The issuer of the time stamp is valid.

**Group**[FRSIGVerifySignatureStates](#)**Head file reference**

fr\_docExpT.h: 267

**FR\_SIG\_TIMESTAMP\_NONE****Syntax**

#define FR\_SIG\_TIMESTAMP\_NONE 0x00400000

**Description**

No timestamp.

**Group**[FRSIGVerifySignatureStates](#)**Head file reference**

fr\_docExpT.h: 255

**FR\_SIG\_TIMESTAMP\_VALID****Syntax**

#define FR\_SIG\_TIMESTAMP\_VALID 0x01000000

**Description**

The time stamp is valid.

**Group**[FRSIGVerifySignatureStates](#)**Head file reference**

fr\_docExpT.h: 259

## FR\_SIG\_VERIFY\_CHANGE

### Syntax

```
#define FR_SIG_VERIFY_CHANGE 0x00000005
```

### Description

The document is changed beyond the limits.

### Group

[FRSIGVerifySignatureStates](#)

### Head file reference

fr\_docExpT.h: 237

## FR\_SIG\_VERIFY\_ERRORBYTERANGE

### Syntax

```
#define FR_SIG_VERIFY_ERRORBYTERANGE 0x00000004
```

### Description

The signature data bytes is not in the expected range.

### Group

[FRSIGVerifySignatureStates](#)

### Head file reference

fr\_docExpT.h: 235

## FR\_SIG\_VERIFY\_ERRORDATA

### Syntax

```
#define FR_SIG_VERIFY_ERRORDATA 0x00000002
```

### Description

The signature data is corrupted and can not be parsed successfully.

### Group

[FRSIGVerifySignatureStates](#)

### Head file reference

fr\_docExpT.h: 231

## FR\_SIG\_VERIFY\_INcredible

### Syntax

```
#define FR_SIG_VERIFY_INcredible 0x00000006
```

**Description**

The signature is not trusted.

**Group**

[FRSIGVerifySignatureStates](#)

**Head file reference**

fr\_docExpT.h: 239

## FR\_SIG\_VERIFY\_INVALID

**Syntax**

```
#define FR_SIG_VERIFY_INVALID 0x00000001
```

**Description**

The signature state is invalid.

**Group**

[FRSIGVerifySignatureStates](#)

**Head file reference**

fr\_docExpT.h: 229

## FR\_SIG\_VERIFY\_ISSUER\_CURRENT

**Syntax**

```
#define FR_SIG_VERIFY_ISSUER_CURRENT 0x00200000
```

**Description**

The issuer is the current issuer.

**Group**

[FRSIGVerifySignatureStates](#)

**Head file reference**

fr\_docExpT.h: 252

## FR\_SIG\_VERIFY\_ISSUER\_EXPIRE

**Syntax**

```
#define FR_SIG_VERIFY_ISSUER_EXPIRE 0x00080000
```

**Description**

The certificate is expired.

**Group**

[FRSIGVerifySignatureStates](#)**Head file reference**

fr\_docExpT.h: 248

**FR\_SIG\_VERIFY\_ISSUER\_REVOKE****Syntax**

#define FR\_SIG\_VERIFY\_ISSUER\_REVOKE 0x00040000

**Description**

The certificate is revoked.

**Group**[FRSIGVerifySignatureStates](#)**Head file reference**

fr\_docExpT.h: 246

**FR\_SIG\_VERIFY\_ISSUER\_UNCHECK****Syntax**

#define FR\_SIG\_VERIFY\_ISSUER\_UNCHECK 0x00100000

**Description**

The issuer is unchecked.

**Group**[FRSIGVerifySignatureStates](#)**Head file reference**

fr\_docExpT.h: 250

**FR\_SIG\_VERIFY\_ISSUER\_UNKNOW****Syntax**

#define FR\_SIG\_VERIFY\_ISSUER\_UNKNOW 0x00020000

**Description**

The issuer is unknown.

**Group**[FRSIGVerifySignatureStates](#)**Head file reference**

fr\_docExpT.h: 244

## FR\_SIG\_VERIFY\_ISSUER\_VALID

### Syntax

```
#define FR_SIG_VERIFY_ISSUER_VALID 0x00010000
```

### Description

The issuer is valid.

### Group

[FRSIGVerifySignatureStates](#)

### Head file reference

fr\_docExpT.h: 242

## FR\_SIG\_VERIFY\_NOSUPPORTWAY

### Syntax

```
#define FR_SIG_VERIFY_NOSUPPORTWAY 0x00000003
```

### Description

The signature type is not supported.

### Group

[FRSIGVerifySignatureStates](#)

### Head file reference

fr\_docExpT.h: 233

## FR\_SIG\_VERIFY\_VALID

### Syntax

```
#define FR_SIG_VERIFY_VALID 0x00000000
```

### Description

The signature state is valid.

### Group

[FRSIGVerifySignatureStates](#)

### Head file reference

fr\_docExpT.h: 227

## FR\_ST\_ANNOTATION

### Syntax

```
#define FR_ST_ANNOTATION "Annotation"
```

**Description**

The selection type of *Annotation*

**Group**

[SelectionType](#)

**Head file reference**

fr\_appExpT.h: 3707

## FR\_ST\_BITMAP

**Syntax**

```
#define FR_ST_BITMAP "Bitmap"
```

**Description**

The selection type of *Bitmap*

**Group**

[SelectionType](#)

**Head file reference**

fr\_appExpT.h: 3705

## FR\_ST\_BOOKMARK

**Syntax**

```
#define FR_ST_BOOKMARK "Bookmark"
```

**Description**

The selection type of *Bookmark*

**Group**

[SelectionType](#)

**Head file reference**

fr\_appExpT.h: 3711

## FR\_ST\_TEXT

**Syntax**

```
#define FR_ST_TEXT "Text"
```

**Description**

The selection type of *Text*

**Group**

[SelectionType](#)**Head file reference**

fr\_appExpT.h: 3703

**FR\_ST\_THUMBNAIL****Syntax**

#define FR\_ST\_THUMBNAIL "Thumbnail"

**Description**The selection type of *Thumbnail***Group**[SelectionType](#)**Head file reference**

fr\_appExpT.h: 3709

**FR\_TASKPANE\_ADVSEARCH****Syntax**

#define FR\_TASKPANE\_ADVSEARCH "AdvSearchTaskPane"

**Description**

The name of advanced search task pane.

**Group**[TaskPaneName](#)**Head file reference**

fr\_appExpT.h: 4475

**Enumerations****Enumerations summary**[FR\\_LOG\\_LEVEL](#)

The definitions of log levels.

[FR\\_SG\\_HANDLE](#)

The handle of the result.

[FR\\_SG\\_ICONTYPE](#)

The icon type of showing graphic on signature appearance.

[FR\\_SG\\_PERMISSION](#)

The definition for signature permission.

[FR\\_SG\\_TEXTDIR](#)

The text direction on signature appearance.

**FRDirectoryType**

Directory type enumeration used to load library. See [FRAppLoadLibrary](#) .

**FRMSGIMPORTANCE**

The importance type of the bulb message.

**FRPDFA\_PDFVersion****FRRibbonStyleButtonType**

The ribbon style control types.

**FRRibbonStyleFlatType**

The flat type.

**FRRibbonStyleImageSide**

The image side.

**FRRibbonStyleTextAlignType**

The alignment type of the text.

**FRStatusBarLocation**

The plug-in can add a window to the status bar in the specified location.

**FRSubscriptionFlowName**

The name definition of subscription flow.

## Enumerations detail

### FR\_LOG\_LEVEL

**Syntax**

```
enum FR_LOG_LEVEL{  
    FR_INFO,  
    FR_WARNING,  
    FR_ERROR,  
    FR_FATAL  
};
```

**Description**

The definitions of log levels.

**Head file reference**

fr\_appExpT.h: 6114

**FR\_INFO**

The info level.

**FR\_WARNING**

The warning level.

**FR\_ERROR**

The error level.

**FR\_FATAL**

### FR\_SG\_HANDLE

**Syntax**

---

```
enum FR_SG_HANDLE{
    FR\_SIG\_HANDLE\_SUCCESS,
    FR\_SIG\_HANDLE\_FAIL,
    FR\_SIG\_HANDLE\_NONE /* None.*/
};
```

**Description**

The handle of the result.

**Head file reference**

fr\_docExpT.h: 275

**FR\_SIG\_HANDLE\_SUCCESS**

Succeed.

**FR\_SIG\_HANDLE\_FAIL**

Fail.

**FR\_SIG\_HANDLE\_NONE /\* None.\*/****FR\_SG\_ICONTYPE****Syntax**

```
enum FR_SG_ICONTYPE{
    FR\_SGIT\_NONE,
    FR\_SGIT\_GRAPHICS,
    FR\_SGIT\_NAME /* The type is name.*/
};
```

**Description**

The icon type of showing graphic on signature appearance.

**Head file reference**

fr\_docExpT.h: 320

**FR\_SGIT\_NONE**

None.

**FR\_SGIT\_GRAPHICS**

The type is graphic.

**FR\_SGIT\_NAME /\* The type is name.\*/****FR\_SG\_PERMISSION****Syntax**

```
enum FR_SG_PERMISSION{
    FR\_APG\_NONE,
    FR\_APG\_ONE,
    FR\_APG\_TWO,
    FR\_APG\_THREE,
    FR\_APG\_LOCK /* Lock document.*/
};
```

```
};
```

**Description**

The definition for signature permission.

**Head file reference**

fr\_docExpT.h: 345

**FR\_APG\_NONE**

None.

**FR\_APG\_ONE**

No changes allowed.

**FR\_APG\_TWO**

Form fill-in and digital signatures.

**FR\_APG\_THREE**

Annotations, form fill-in, and digital signatures.

**FR\_APG\_LOCK /\*\* Lock document. \*/****FR\_SG\_TEXDIR****Syntax**

```
enum FR_SG_TEXDIR{
    FR\_SGTD\_AUTO,
    FR\_SGTD\_LEFTTORIGHT,
    FR\_SGTD\_RIGHTTOLEFT /** Right to left. */
};
```

**Description**

The text direction on signature appearance.

**Head file reference**

fr\_docExpT.h: 332

**FR\_SGTD\_AUTO**

Auto.

**FR\_SGTD\_LEFTTORIGHT**

Left to right.

**FR\_SGTD\_RIGHTTOLEFT /\*\* Right to left. \*/****FRDirectoryType****Syntax**

```
enum FRDirectoryType{
    FR\_DIRECTORY\_PATH\_SYSTEM,
    FR\_DIRECTORY\_PATH\_APPLICATION,
};
```

```
FR_DIRECTORY_PATH_PLUGIN,  
FR_DIRECTORY_PATH_USERSPECIFY /*Loads the library from the path  
that the user specified. */  
};
```

**Description**

Directory type enumeration used to load library. See [FRApplLoadLibrary](#) .

**Head file reference**

fr\_appExpT.h: 5275

**FR\_DIRECTORY\_PATH\_SYSTEM**

Loads the library from the system path.

**FR\_DIRECTORY\_PATH\_APPLICATION**

Loads the library from the application path.

**FR\_DIRECTORY\_PATH\_PLUGIN**

Loads the library from the plug-in path.

**FR\_DIRECTORY\_PATH\_USERSPECIFY /\*Loads the library from the path that  
the user specified. \*/****FRMSGIMPORTANCE****Syntax**

```
enum FRMSGIMPORTANCE{  
    FR_IMPO_LOW,  
    FR_IMPO_HIGH  
};
```

**Description**

The importance type of the bulb message.

**Head file reference**

fr\_barExpT.h: 1527

**FR\_IMPO\_LOW**

The importance is low.

**FR\_IMPO\_HIGH**

The importance is high.

**FRPDFA\_PDFVersion****Syntax**

```
enum FRPDFA_PDFVersion{ };
```

**Description**

**Head file reference**

fr\_appExpT.h: 6040

**FRRibbonStyleButtonType****Syntax**

```
enum FRRibbonStyleButtonType{
    FR\_RibbonStyle\_Button,
    FR\_RibbonStyle\_Edit,
    FR\_RibbonStyle\_CheckBox,
    FR\_RibbonStyle\_RadioButton,
    FR\_RibbonStyle\_SpinButtonCtrl,
    FR\_RibbonStyle\_SliderCtrl,
    FR\_RibbonStyle\_LinkButton,
    FR\_RibbonStyle\_ComboGalleryCtrl,
    FR\_RibbonStyle\_GalleryCtrl,
    FR\_RibbonStyle\_Static,
    FR\_RibbonStyle\_ListBox /** The ribbon style list box. */
};
```

**Description**

The ribbon style control types.

**Head file reference**

fr\_barExpT.h: 1350

**FR\_RibbonStyle\_Button**

The ribbon style button.

**FR\_RibbonStyle>Edit**

The ribbon style edit box.

**FR\_RibbonStyle\_CheckBox**

The ribbon style check box.

**FR\_RibbonStyle\_RadioButton**

The ribbon style radio box.

**FR\_RibbonStyle\_SpinButtonCtrl**

The ribbon style spin button.

**FR\_RibbonStyle\_SliderCtrl**

The ribbon style slider.

**FR\_RibbonStyle\_LinkButton**

The ribbon style link button.

**FR\_RibbonStyle\_ComboGalleryCtrl**

The ribbon style combo gallery.

**FR\_RibbonStyle\_GalleryCtrl**

The ribbon style gallery.

**FR\_RibbonStyle\_Static**

The ribbon style static.

**FR\_RibbonStyle\_ListBox** /\* The ribbon style list box. \*/**FRRibbonStyleFlatType****Syntax**

```
enum FRRibbonStyleFlatType{  
    FR\_RibbonStyle\_FlatType\_3D,  
    FR\_RibbonStyle\_FlatType\_Flat,  
    FR\_RibbonStyle\_FlatType\_NoBorders /* The flat type with no borders. */  
};
```

**Description**

The flat type.

**Head file reference**

fr\_barExpT.h: 1388

**FR\_RibbonStyle\_FlatType\_3D**

The 3D flat type.

**FR\_RibbonStyle\_FlatType\_Flat**

The normal flat type.

**FR\_RibbonStyle\_FlatType\_NoBorders** /\* The flat type with no borders. \*/**FRRibbonStyleImageSide****Syntax**

```
enum FRRibbonStyleImageSide{  
    FR\_RibbonStyle\_LeftImage,  
    FR\_RibbonStyle\_RightImage,  
    FR\_RibbonStyle\_TopImage  
};
```

**Description**

The image side.

**Head file reference**

fr\_barExpT.h: 1378

**FR\_RibbonStyle\_LeftImage**

Left side.

**FR\_RibbonStyle\_RightImage**

Right side.

**FR\_RibbonStyle\_TopImage**

Top side.

## FRRibbonStyle.TextAlignType

### Syntax

```
enum FRRibbonStyle.TextAlignType{
    FR_RibbonStyle_Align_Left,
    FR_RibbonStyle_Align_Right,
    FR_RibbonStyle_Align_Center /* Center alignment. */
};
```

### Description

The alignment type of the text.

### Head file reference

fr\_barExpT.h: 1368

#### **FR\_RibbonStyle\_Align\_Left**

Left alignment.

#### **FR\_RibbonStyle\_Align\_Right**

Right alignment.

#### **FR\_RibbonStyle\_Align\_Center /\* Center alignment. \*/**

## FRStatusBarLocation

### Syntax

```
enum FRStatusBarLocation{
    FR_STATUSBAR_LOCATION_LEFT,
    FR_STATUSBAR_LOCATION_CENTER,
    FR_STATUSBAR_LOCATION_RIGHT /* The right location. */
};
```

### Description

The plug-in can add a window to the status bar in the specified location.

### Head file reference

fr\_barExpT.h: 1439

#### **FR\_STATUSBAR\_LOCATION\_LEFT**

The left location.

#### **FR\_STATUSBAR\_LOCATION\_CENTER**

The center location.

#### **FR\_STATUSBAR\_LOCATION\_RIGHT /\* The right location. \*/**

## FRSubscriptionFlowName

**Syntax**

```
enum FRSubscriptionFlowName{
    FR\_SUBSCRIBE,
    FR\_EXTEND\_SUBSCRIBE,
    FR\_SUBSCRIB\_MANAGE,
    FR\_EXTEND\_SUBSCRIBE\_AUTO /\*Extend the subscription auto. \*/
};
```

**Description**

The name definition of subscription flow.

**Head file reference**

fr\_appExpT.h: 5953

**FR\_SUBSCRIBE**

Subscribe.

**FR\_EXTEND\_SUBSCRIBE**

Extend the subscription.

**FR\_SUBSCRIB\_MANAGE**

Manage the subscription.

**FR\_EXTEND\_SUBSCRIBE\_AUTO /\*Extend the subscription auto. \*/**

## Callbacks

### Callbacks summary

**FRBulbMsgCheckCallback**

The callback is invoked by Foxit Reader when user clicks the check box of the bulb message. The check box is used to check that whether to not show the bulb message again.

**FRBulbMsgOperationCallback**

The callback is invoked by Foxit Reader when user clicks the operation button of the bulb message.

**FRComputeEnabledProc**

A callback that is used to determine whether a menu item, toolbar button is enabled.

**FRComputeMarkedProc**

A callback that is used to determine whether a menu item or toolbar button is marked (a marked menuitem has a check mark next to it, and a marked toolbar button appears selected).

**FRDropDownProc**

The callback is invoked by Foxit Reader when user clicks the ribbon element drop-down arrow

**FRExecuteProc**

A callback that is called whenever a menu item, toolbar button or ribbon element is executed. It implements whatever themenu item, toolbar button or ribbon element does.

**FRFreeDataProc**

A callback that is used to free data.

**FRGetMessageStringProc**

A callback that is called whenever a menu is popped up to receive the tooltip.

**FRPopUpMenuProc**

A callback that is called when the context menu is popped up in the doc view.

**FRRibbonElementImageInitProc**

The callback is invoked by Foxit Reader when the ribbon object is to init the image.

**FRRibbonPaletteButtonHighlightItemProc**

The callback is invoked by Foxit Reader when the mouse is over ribbon palette object.

**FRRibbonPanelImageInitProc**

The callback is invoked by Foxit Reader when the ribbon object is to init the image.

**FRActionHandlerDocOpen**

It is called by Foxit Reader if the document contains an action to be performed when the document is opened.

**FRActionHandlerJavaScript**

It is called by Foxit Reader if the document contains a JavaScript action to be performed when the document is opened.

**FRActionHandlerPage**

It is called by Foxit Reader if the page contains an additional-actions dictionary defining actions to be performedwhen the page is opened or closed.

**FRActionHandlerLink**

It is called by Foxit Reader if the link annotation contains an actions to be performed when the annotation is activated.

**FRAppOnLangUIChange**

A callback for application level event handler.

**FRAppOnActivate**

A callback for application level event handler.

**FRAppWillQuit**

A callback for application level event handler.

**FRAppWillQuit**

A callback for application level event handler.

**FRAppOnToolChanged**

A callback for application level event handler.

**FRAppOnSkinChange**

A callback for application level event handler.

**FRAppOnShowFullScreen**

A callback for application level event handler.

**FRAppOnHideFullScreen**

A callback for application level event handler.

**FRAppOnResetToolbars**

A callback for application level event handler.

**FRAppOnRunCommandLine**

A callback for application level event handler.

**FRAppOnDrop**

A callback for application level event handler.

**FRAppOnTextViewChange**

A callback for application level event handler.

**FRAppOnRecentFileListChange**

A callback for application level event handler.

**FRAppOnShowRibbonFilePage**

A callback for application level event handler.

**FRAppOnCloseRibbonFilePage**

A callback for application level event handler.

**FRAppOnRibbonElementsLoadFinish**

A callback for application level event handler.

**FRApOnOpenDocument**

A callback for application level event handler.

**FRApOnRibbonCategoryClicked**

A callback for application level event handler.

**FRApOnMainFrmLoadFinish**

A callback for application level event handler.

**FRApOnRibbonUILayoutFinish**

A callback for application level event handler.

**FRApOnShowTaskPane**

A callback for application level event handler.

**FRApOnShowPopupMenu**

A callback for application level event handler.

**FRApOnDidOpenAllFiles**

A callback for application level event handler.

**FRApOnMainFrameWillClose**

A callback for application level event handler.

**FRApMainFrameOnSize**

A callback for application level event handler.

**FRApMainFrameOnLoadWinPlacementFinish**

A callback for application level event handler.

**FRApPluginOnLoaded**

A callback for application level event handler.

**FRApOnDidCloseRibbonFilePage**

A callback for application level event handler.

**FRCaptureGetType**

It is called to get the capture type. See the group definition [CaptureType](#) .

**FRCaptureLosingCapture**

It is called by Foxit Reader to notify that the current capture handler is losing capture.

**FRCaptureGettingCapture**

It is called by Foxit Reader to notify that the current capture handler is getting capture.

**FRCaptureLButtonDown**

It is called by Foxit Reader to notify the left-button-down event.

**FRCaptureLButtonUp**

It is called by Foxit Reader to notify the left-button-up event.

**FRCaptureLButtonDblClk**

It is called by Foxit Reader to notify the left-button-DblClk event.

**FRCaptureMouseMove**

It is called by Foxit Reader to notify the mouse-move event.

**FRCaptureRButtonDown**

It is called by Foxit Reader to notify the right-button-down event.

**FRCaptureRButtonUp**

It is called by Foxit Reader to notify the right-button-up event.

**FRCaptureRButtonDblClk**

It is called by Foxit Reader to notify the right-button-DblClk event.

**FRCaptureContextMenu**

It is called by Foxit Reader to notify the context menu pop-up event.

**FRConProviderOnFileOpen**

It will be invoked by Foxit Reader when a document is opened. If it returns [TRUE](#) ,it means the plug-in can process the document and the other callbacks go on. Otherwise not.

**FRConProviderOnGetPermissions**

The plug-in can control the permissions of the document.

**FRConProviderOnGetContentSize**

It will be invoked by Foxit Reader to receive the size of the document the plug-in provides.

**FRConProviderOnReadContent**

It will be invoked by Foxit Reader to receive the the document data the plug-in provides.This routine occurs times without number and the data is read in blocks.

**FRConProviderOnWriteContent**

It will be invoked by Foxit Reader when the document is modified and is saved. The plug-in must re-encryptthe document and save the document in some format.

**FRConProviderOnFileClose**

It will be invoked by Foxit Reader when the document is closed.

**FRConProviderOnWriteAttachmentContent**

When the attachment is opened, it will be saved to a temp directory first. This interface leaves the choice of protectingthe attachment to the plug-in. You can get the type of the attachment from the file path.

**FRConProviderOnGetAttachmentSize**

When the attachment is modified, the Foxit Reader needs to reattach the attachment. This interface is invoked toget the total size of the attachment modified.

**FRConProviderOnReadAttachmentContent**

When the attachment is modified, the Foxit Reader needs to reattach the attachment. This interface is invoked toget the content of the attachment modified. Since

SDK\_LATEEST\_VERSION > 3.0, FRConProviderOnReadAttachmentContent is replaced with FRConProviderOnReadAttachmentContentInBlocks .

**FRConProviderIsPageAvail**

It will be invoked by Foxit Reader when a PDF page data can not be obtained and the PDF page will be rendered.

**FRConProviderNeedReopenDoc****FRConProviderCanBeSaved****FRContentProviderIsProcessErrMsg**

Checks whether the plug-in processes the error message.

**FRContentProviderIsSupportPanel**

Checks Whether the document supports panel.

**FRContentProviderIsSupportViewByScroll**

Checks whether the document supports viewing by scrolling.

**FRContentProviderIsSupportViewByScroll**

Determine whether to delete the navigation panel.

**FROnGetPageDictObjectNumber**

This interface is reserved.

**FRConProviderIsWriteContentProgressive**

It will be invoked by Foxit Reader to check whether write the content progressively or all at once.

**FRConProviderWriteContentProgressiveFinish**

It will be invoked by Foxit Reader to notify that finish writing content progressively.

**FRConProviderIsWriteAttachmentContent**

When the attachment is opened, it will be saved to a temp directory first. This interface will be invoked to determine that whetherthe plug-in is to write the attachment content itself or not. If this interface returns TRUE, FRConProviderOnWriteAttachmentContent willbe invoked several times to write the attachment content in blocks.

**FRConProviderOnWriteAttachmentContentFinish**

When the attachment is opened, it will be saved to a temp directory first. This interface will be invoked when finishing writingthe attachment content to the temp directory.

**FRConProviderOnReadAttachmentContentInBlocks**

When the attachment is modified, the Foxit Reader needs to reattach the attachment. This interface is invoked toget the content of the attachment modified. Firstly,

[\*\*FRConProviderOnGetAttachmentSize\*\*](#) will be invoked to obtain the size of the attachment content. Since [SDK LATEST VERSION](#) > 3.0,  
[\*\*FRConProviderOnReadAttachmentContent\*\*](#) is replaced with  
[\*\*FRConProviderOnReadAttachmentContentInBlocks\*\*](#) .

**[FRConProviderOnBackFillContent](#)**

It will be invoked by Foxit Reader when some data need to be backfilled after [\*\*FRConProviderOnWriteAttachmentContentFinish\*\*](#) finishes.

**[FREnryptCreateHandler](#)**

Initializes the crypto handler with the data of security handler, and the encrypt dictionary data.

**[FREnryptDecryptGetSize](#)**

Estimate size of decrypted data, from a source size. If implementation doesn't want to estimate, it can always return 0.

**[FREnryptDecryptStart](#)**

Starts a decryption process.

**[FREnryptDecryptStream](#)**

Decrypt some source data in a stream.

**[FREnryptDecryptFinish](#)**

Finish a decryption process.

**[FREnryptEncryptGetSize](#)**

Get encrypted data size for a source data block. The returned size should equal to or be larger than the final encrypted data block.

**[FREnryptEncryptContent](#)**

Do the encryption process. Final encrypted data block should be output in the *outDestsize* parameter.

**[FREnryptFinishHandler](#)**

Release the crypto handler.

**[FREnryptProgressiveEncryptStart](#)**

It is invoked by Foxit Reader when encrypting if you set the progressive encrypt handler by [\*\*FPDCreatorSetProgressiveEncryptHandler\*\*](#).

**[FREnryptProgressiveEncryptContent](#)**

It is invoked by Foxit Reader over and over when encrypting if you set the progressive encrypt handler by [\*\*FPDCreatorSetProgressiveEncryptHandler\*\*](#). Implementation should append the encrypted data (if any) to the *dest\_buf* dynamic array.

**[FREnryptProgressiveEncryptFinish](#)**

It is invoked by Foxit Reader when finish encrypting content progressively if you set the progressive encrypt handler by [\*\*FPDCreatorSetProgressiveEncryptHandler\*\*](#). Implementation should append the encrypted data (if any) to the *dest\_buf* dynamic array.

**[FROnCollectNormalData](#)**

It is called by Foxit Reader when some action occurs.

**[FROnCollectBitmapData](#)**

It is called by Foxit Reader when some action occurs.

**[FROnCollectNormalData2](#)**

It is called by Foxit Reader when some action occurs.

**[FROnCollectBitmapData2](#)**

It is called by Foxit Reader when some action occurs.

**[FRDocPropertyPageOnCreate](#)**

A callback for document-property-page handler. It is called after the property dialog is created. Client should create a new property page and register it using [\*\*FRAppAddDocPropertyPage\*\*](#) () in this method.

**[FRDocPropertyPageOnDestroy](#)**

A callback for document-property-page handler. It is called when user closes the property dialog. Client should destroy the property page window and free dynamic memory in this method.

**FRDocPropertyPageOnSaveData**

A callback for document-property-page handler. It is called when user click the *OK* button on the property dialog.

**FRCryptoSetKey**

Set the key when encryption and decryption.

**FRCryptoCurrentEncryptObjNum**

Get current encryption PDF object number.

**FRSecurityOnInit**

It is invoked to initialize the security handler.

**FRSecurityGetPermissions**

Create a crypto handler that can do the real encryption/decryption work.

**FRSecurityIsProcessErrMsg**

Checks whether the plug-in processes the error message.

**FRAppEmptyFrameWndCanClose**

It is called when user click the "close" button to close a frame which created by `FRAppCreateAnEmptyFrameWnd ()`.

**FRTоМаст**

Tests whether a document can be controlled to print.

**FRCCanBePrinted**

Tests whether a document can be printed.

**FRCountOwnerTextPrintInfo**

Gets the total number of text information which will be printed as a watermark of specified page.

**FRCountOwnerImgPrintInfo**

Gets the total number of image information which will be printed as a watermark of specified page.

**FRPrintOwnerTextInfo**

Passes the text information to Foxit Reader to print as a watermark of a page.

**FRPrintOwnerImgInfo****FRPrintOnPreviewDidRender**

It is invoked by Foxit Reader when the page has been rendered to preview dialog. You can use the *renderDevice* to render extra content to the preview dialog.

**FRPrintOnDidRender**

It is invoked by Foxit Reader when the page has been printed to the printer. You can use the *renderDevice* to print extra content to the printer.

**FRPrintIsPrintOpacity**

It is invoked by Foxit Reader before the page is rendered to memory device.

**FRCCanBePrinted2**

Tests whether a document can be printed.

**FRCaptureGetType**

It is called to get the mouse point handler type.

**FRMousePtGetObjectAtPoint**

It is called to get the object at the specified point.

**FRMousePtLButtonDown**

It is called by Foxit Reader to notify the left-button-down event.

**FRMousePtLButtonUp**

It is called by Foxit Reader to notify the left-button-up event.

**FRMousePtLButtonDblClk**

It is called by Foxit Reader to notify the left-button-double-click event.

**FRMousePtMouseMove**

It is called by Foxit Reader to notify the mouse-move event.

**FRMousePtRButtonDown**

It is called by Foxit Reader to notify the right-button-down event.

**FRMousePtRButtonUp**

It is called by Foxit Reader to notify the right-button-up event.

**FRMousePtRButtonDblClk**

It is called by Foxit Reader to notify the right-button-double-click event.

**FRMousePtMouseWheel**

It is called by Foxit Reader to notify the mouse-wheel event.

**FRMousePtOnMouseEnter**

It is called by Foxit Reader to notify the mouse-enter event.

**FRMousePtOnMouseExit**

It is called by Foxit Reader to notify the mouse-exit event.

**FROwnerFileTypeHandlerDoOpen**

It will be invoked by Foxit Reader when a document is opened. You can implement the callback to control the process of opening.

**FROwnerFileTypeHandlerDoSave**

It will be invoked by Foxit Reader when a document is saved. You can implement the callback to control the process of saving.

**FROwnerFileTypeHandlerDoEmail**

It will be invoked by Foxit Reader when a document is emailed. You can implement the callback to control the process of sending email.

**FROwnerFileTypeHandlerCanSetFileAssociation**

It will be invoked by Foxit Reader to judge whether the plug-in need to associate the file type with Foxit Reader.

**FROwnerFileTypeHandlerSetFileAssociationInfo**

It will be invoked by Foxit Reader to set the file association information.

**FROwnerFileTypeHandlerGetSupportFileTypeCount**

It will be invoked by Foxit Reader to get the count of file types that this handler can support.

**FROwnerFileTypeHandlerGetFileFilterNameByIndex**

It will be invoked by Foxit Reader to get the filter name of custom file type by index.

**FROwnerFileTypeHandlerGetFileTypeFilter**

It will be invoked by Foxit Reader to get the filter of the custom file type.

**FROwnerFileTypeHandlerGetFileExt**

It will be invoked by Foxit Reader to get the extension of the custom file type.

**FROwnerFileTypeHandlerIsEnableSaveAsSettingBtn**

It will be invoked by Foxit Reader to determine whether enables the setting button in the Save As dialog.

**FROwnerFileTypeHandlerExecuteSaveAsSetting**

It will be invoked by Foxit Reader when the user clicks the setting button.

**FROwnerFileTypeHandlerCanSupportAddToRecentList**

It will be invoked by Foxit Reader to determine whether the file of this type can be added to the recent file list.

**FROwnerFileTypeHandlerCanSupportSave**

It will be invoked by Foxit Reader to determine whether the filter of the document can support to save.

**FROwnerFileTypeHandlerCanBeSaveAsToOtherExt**

It will be invoked by Foxit Reader to determine whether the filter can be save as to other extensions.

**FROwnerFileTypeHandlerCanSupportOpen**

It will be invoked by Foxit Reader to determine whether the filter of the document can support to open.

**FROwnerFileTypeHandlerDoOpenDir**

It will be invoked by Foxit Reader to open a dir which includes the opening epub document.

**FROwnerFileTypeHandlerCanSupportOpenDir**

It will be invoked by Foxit Reader to determine whether the filter of the document can support to open dir for this current epub document.

**FROwnerFileTypeHandlerSaveFilePathToClipboard**

It will be invoked by Foxit Reader to copy current epub document path to clip board.

**FRDocWillOpen**

A callback for document-level event handler.

**FRDocDidOpen**

A callback for document-level event handler.

**FRDocOnActivate**

A callback for document-level event handler.

**FRDocOnDeactivate**

A callback for document-level event handler.

**FRDocWillSave**

A callback for document-level event handler.

**FRDocDidSave**

A callback for document-level event handler.

**FRDocWillClose**

A callback for document-level event handler.

**FRDocDidClose**

A callback for document-level event handler.

**FRDocDidCopy**

A callback for document-level event handler.

**FRDocWillPrint**

A callback for document-level event handler.

**FRDocDidPrint**

A callback for document-level event handler.

**FRDocOnChange**

A callback for document-level event handler.

**FRDocOnPermissionChange**

A callback for document-level event handler.

**FRDocWillDraw**

A callback for document-level event handler.

**FRDocDidDraw**

A callback for document-level event handler.

**FRDocOnWillInsertPages**

A callback for document-level event handler.

**FRDocOnDidInsertPages**

A callback for document-level event handler.

**FRDocOnWillDeletePages**

A callback for document-level event handler.

**DidDeletePages**

A callback for document-level event handler.

**FRDocOnWillModifyPageAttribute**

A callback for document-level event handler.

**FRDocOnDidModifyPageAttribute**

A callback for document-level event handler.

**FRDocOnWindowCreate**

A callback for document-level event handler.

**FRDocOnWindowDestroy**

A callback for document-level event handler.

**FRDocOnFrameCreate**

A callback for document-level event handler.

**FRDocOnFrameDestroy**

A callback for document-level event handler.

**FRDocOnAnnotSelectionChanged**

A callback for document-level event handler.

**FRDocOnAutoScrollBegin**

A callback for document-level event handler.

**FRDocOnAutoScrollEnd**

A callback for document-level event handler.

**FRDocOnFinishRender**

A callback for document-level event handler.

**FRDocThumbnailWillDraw**

A callback for document-level event handler.

**FRDocThumbnailDidDraw**

A callback for document-level event handler.

**FRDocScrollBarThumbnailViewWillDraw**

A callback for document-level event handler.

**FRDocDidFileClose**

A callback for document-level event handler.

**FRDocCanBeSaved**

A callback for document-level event handler.

**OnDocPromptToSave**

A callback for document-level event handler.

**FRDocWillReOpen**

A callback for document-level event handler.

**FRDocDidReOpen**

A callback for document-level event handler.

**FRDocOnFrameSize**

A callback for document-level event handler.

**FRDocOnWillActivate**

A callback for document-level event handler.

**FRDocOnWillDeactivate**

A callback for document-level event handler.

**FRDocOnOtherDocActivate**

A callback for document-level event handler.

**FRDocOnOtherDocDeactivate**

A callback for document-level event handler.

**FRDocOnOtherDocClose**

A callback for document-level event handler.

**FRDocDidSave2**

A callback for document-level event handler.

**FRDocOnKeyDown**

A callback for document-level event handler.

**FRDocOnKeyUp**

A callback for document-level event handler.

**FRDocOnChar**

A callback for document-level event handler.

**FRDocOnLButtonDown**

A callback for document-level event handler.

**FRDocOnLButtonUp**

A callback for document-level event handler.

**FRDocOnLButtonDblClk**

A callback for document-level event handler.

**FRDocOnMouseMove**

A callback for document-level event handler.

**FRDocOnRButtonDown**

A callback for document-level event handler.

**FRDocOnRButtonUp**

A callback for document-level event handler.

**FRDocOnRButtonDblClk**

A callback for document-level event handler.

**FRDocOnMouseWheel**

A callback for document-level event handler.

**FRDocOnDrawAnnot**

A callback for document-level event handler.

**FRDocOnDocCollectActionData**

A callback for document-level event handler.

**FRDocWillOpen2**

A callback for document-level event handler.

**FRDocOnOptimizerFinish**

A callback for document-level event handler.

**FRDocCanBeClose**

A callback for document-level event handler.

**FRDocOnReOpenFailed**

A callback for document-level event handler.

**FRDocWillSave2**

A callback for document-level event handler.

**FRDocSaveAsBeforeReopen**

A callback for document-level event handler.

**FRDocCanPaste**

A callback for document-level event handler.

**OnMouseClickOnText**

A callback for document-level event handler.

**FRDocOwnerSaveAs**

A callback for document-level event handler.

**FRDocOnCanDetache**

A callback for document-level event handler.

**FRDocDelayDidOpen**

A callback for document-level event handler.

**FRDocOnActivate2**

A callback for document-level event handler.

**FRDocOnDeactivate2**

A callback for page-level event handler.

**FRPageViewOnClose**

A callback for page-level event handler.

**FRPageViewOnVisible**

A callback for page-level event handler.

**FRPageViewInvisible**

A callback for page-level event handler.

**FRPageViewOnContentChanged**

A callback for page-level event handler.

**FRPageViewOnWillParsePage**

A callback for page-level event handler.

**FRPageViewOnDidParsePage**

A callback for page-level event handler.

**FRPageViewOnContentChanged2**

A callback for page-level event handler.

**FRPanelViewGetName**

A callback for navigation panel view. It is called by Foxit Reader to get the panel view's name.

**FRPanelViewGetTitle**

A callback for navigation panel view. It is called by Foxit Reader to get the panel view's title which is shown in user interface.

**FRPanelViewInitNewView**

A callback for navigation panel view. It is called when a PDF document is open and to be displayed. The plug-in can init the data of the panel view in this callback. Then create the window attached to the panel view in [FRPanelViewOnPanelActive](#).

**FRPanelViewOnPanelActive**

A callback for navigation panel view. It creates a window to attach to navigation panel. Using the window API function *CreateWindow* or *CreateWindowEx* to create a window for returning.

**FRPanelViewOnActive**

A callback for navigation panel view. It is called when a panel view becomes a active view.

**FRPanelViewOnRotate**

A callback for navigation panel view to rotate. It is called after the page view is rotated. You should do some rotation with the panel view.

**FRPanelViewOnDestroyView**

A callback for navigation panel view. It is called at the shutdown time of panel view to free dynamic memory. The panel view should be destroyed in this method.

**FRPanelViewIsDockToBottom**

A callback for navigation panel view. It is called to tell Foxit Reader whether the panel view need to dock bottom.

**FRPanelViewGetButtonTip**

A callback for navigation panel view to pass the tip of panel button. Gets the tip of panel button that displaying on the left bar of panel.

**FRPanelViewGetButtonDescribeText**

A callback for navigation panel view to tell the describe text. Gets the describe text of panel view. The describe text will be displayed on the left corner of status bar when the mouse over the panel button. This interface is not available in version 1.0.

**FRPanelViewGetButtonIcon**

A callback for navigation panel view to get the icon of panel button. The bitmap returned by this method will be taken by Reader, client can not release it until user exit Reader.

**FRPanelViewSetPos**

A callback for navigation panel view to locate the navigation item(such as bookmark item). It is called when user scroll a page view or go to other pages.

**FRPDFASaveAsPDFA**

**FRPOOnBeforeInsertPages**

A callback for page organizing event handler.

**FRPOOnDoInsertPagesDictFinish**

A callback for page organizing event handler.

**FRPOOnAfterInsertPages**

A callback for page organizing event handler.

**FRPOOnBeforeDeletePages**

A callback for page organizing event handler.

**FRPOOnAfterDeletePages**

A callback for page organizing event handler.

**FRPOOnBeforeReplacePages**

A callback for page organizing event handler.

**FRPOOnAfterReplacePages**

A callback for page organizing event handler.

**FRPOOnBeforeSwapTwoPage**

A callback for page organizing event handler.

**FRPOOnAfterSwapTwoPage**

A callback for page organizing event handler.

**FRPOOnBeforeRotatePage**

A callback for page organizing event handler.

**FRPOOnAfterRotatePage**

A callback for page organizing event handler.

**FRPOOnBeforeResizePage**

A callback for page organizing event handler.

**FRPOOnAfterResizePage**

A callback for page organizing event handler.

**FRPOOnBeforeExtractPage**

A callback for page organizing event handler.

**FRPOOnAfterExtractPage**

A callback for page organizing event handler.

**FRPOOnBeforeModifyPageAttr**

A callback for page organizing event handler.

**FRPOOnAfterModifyPageAttr**

A callback for page organizing event handler.

**FRPOOnBeforeMovePages**

A callback for page organizing event handler.

**FRPOOnAfterMovePages**

A callback for page organizing event handler.

**FRPOOnRelease**

A callback for page organizing event handler.

**FRPrefPageOnCreate**

A callback for Reader preference page. It is called after the preference dialog is created.

Client should create a new preference page and register it using [FRAppAddPreferencePage](#) () in this method.

**FRPrefPageOnDestroy**

A callback for Reader preference page. It is called when user closes the preference dialog.

Client should destroy the preference page window and free dynamic memory in this method.

**FRPrefPageOnSaveData**

A callback for Reader preference page. It is called when user clicks the *OK* button on the property dialog.

**FRPrefPageOnGetTabTitle**

It is called by Foxit Reader to get the title of the preference page tab.

**FRRibbonRecentFileChangItemPinned**

A callback for Ribbon recent file list event handler.

**FRRibbonRecentFileRemoveItem**

A callback for Ribbon recent file list event handler.

**FRSecurityOnInit**

It is invoked to initialize the security handler.

**FRSecurityIsProcessErrMsg**

Checks whether the plug-in processes the error message.

**FRSecurityGetPermissions**

Gets permission settings of the document.

**FRSecurityIsOwner**

Checks whether the current user is owner of the document.

**FRSecurityGetCryptInfo**

Get encryption information including standard algorithm and key.

**FRSecurityIsMetadataEncrypted**

Checks if document metadata needs to be encrypted.

**FRSecurityFinishHandler**

Release the security handler.

**FRSecurityCreateCryptoHandler**

Create a crypto handler that can do the real encryption/decryption work. After creation, the caller should call [FREncryptCreateHandler](#) () to initialize the crypto handler.

**FRSecurityMethodGetName**

It is called by Foxit Reader to get the name of the security method.

**FRSecurityMethodGetTitle**

It is called by Foxit Reader to get the title of the security method.

**FRSecurityMethodIsMyMethod**

It is called by Foxit Reader to ask the plug-in whether the security applied for the document can be handled.

**FRSecurityMethodCheckModuleLicense**

It is called by Foxit Reader to check whether the plug-in has the license to modify the security. In general, it is a internal interface.

**FRSecurityMethodCanBeModified**

It is called by Foxit Reader to check whether the plug-in has the right to modify the security.

**FRSecurityMethodDoSetting**

It is called by Foxit Reader when the user clicks the setting button in the security page of document property dialog. You can implement your own setting process.

**FRSecurityMethodRemoveSecurityInfo**

It is called by Foxit Reader when the security method is to be removed. Then you can implement your own removing process.

**FRSecurityMethodCreatePermSubDlg**

It is called by Foxit Reader when the security page of document property dialog is to be shown. If the security method is yours, you have to create a child window to show the information of the security method.

**FRSecurityMethodDestroyPermSubDlg**

It is called by Foxit Reader when the security page of document property dialog is to be destroyed.

**FRSelectionGetType**

It returns the selection type (for example, 'Text' or 'Bookmark'). This information is used so that the Foxit Reader knows which selection handler to call.

**FRSelectionCanSelectAll**

It is used to determine whether the current selection type can perform a select all operation. This controls whether the Select All menu item is enabled.

**FRSelectionDoSelectAll**

It is called to perform the select all operation.

**FRSelectionCanDelete**

It is used to determine whether the current selection can be deleted. This controls, for example, whether the *Delete* menu item is enabled.

**FRSelectionDoDelete**

It deletes the current selection.

**FRSelectionCanCopy**

It is used to determine whether the current selection can be copied. This controls, for example, whether the *Copy* menu item is enabled.

**FRSelectionDoCopy**

It copies the selected item to the clipboard.

**FRSelectionCanCut**

It is used to determine whether the current selection can be cut. This controls, for example, whether the *Cut* menu item is enabled.

**FRSelectionDoCut**

It cuts the current selection. See the discussion under [FRSelectionDoCopy\(\)](#) for information on how the selection handler must use the clipboard.

**FRSelectionCanPaste**

It is used to determine whether the current selection can be pasted. This controls, for example, whether the *Paste* menu item is enabled.

**FRSelectionDoPaste**

It pastes the current selection from the clipboard.

**FRSelectionLosingSelection**

This method is called by [FRSelectionClearSelection](#) (among others), to let the selection handler responsible for the old selection do whatever cleanup it needs.

**FRSelectionGettingSelection**

It is called when the selection is set (for example, via FRSelectionSetCurSelection()).

**FRSelectionRemovedFromSelection**

A callback that de-highlights the old item given in remData, and returns a new curSelectData or [NULL](#) if failure occurred.

**FRSelectionAddedToSelection**

A callback that adds the specified item to the selection, highlights it, and returns the new selection containing the newly-added item.

**FRSelectionShowSelection**

It changes the view (for example, by scrolling the current page or moving to the appropriate page) so that the current selection is visible.

**FRSelectionKeyDown**

It handles a key press. It is needed only if the selection handler processes key presses.

**FRSelectionKeyUp**

It handles a key press. It is needed only if the selection handler processes key presses.

**FRSelectionKeyChar**

It handles a key press. It is needed only if the selection handler processes key presses.

**FRSelectionMouseWheel**

It handles a mouse wheeling. It is needed only if the selection handler processes mouse wheeling.

**FRSelectionCanDeselectAll**

It is used to determine whether all the current selections can be deselected. This controls, for example, whether the *Deselect All* menu item is enabled.

**FRSelectionDoDeselectAll**

It deselects all the items selected.

**FRSignatureHandlerGetName**

A callback for signature handler.

**FRSignatureHandlerSignData**

A callback for signature handler.

**FRSignatureHandlerTimeStampDate**

A callback for signature handler.

**FRSignatureHandlerVerifyData**

A callback for signature handler.

**FRSignatureHandlerShowStateUI**

A callback for signature handler.

**FRSignatureHandlerCanClear**

A callback for signature handler.

**FRSignatureHandlerShowSignProperties**

A callback for signature handler.

**RSignatureHandlerReleaseSignData**

A callback for signature handler.

**FRStatusBarWndExOnGetTooltip**

A callback for adding a window to the status bar.

**FRStatusBarWndExOnCreateWnd**

A callback for adding a window to the status bar.

**FRStatusBarWndExGetSize**

A callback for adding a window to the status bar.

**FRStatusBarWndExOnSize**

A callback for adding a window to the status bar.

**FRShowSubscribeRibbonUI**

A callback for subscription provider.

**FRStartSubscribeWorkflow**

A callback for subscription provider.

**FRIsLicenseRevoked**

A callback for subscription provider.

**FRShowSubscribeFlash**

A callback for subscription provider.

**OnUndo**

It is called to implement the undo operation.

**OnRedo**

It is called to implement the redo operation.

**OnRelease**

It is called when the undo-redo item is to be released by Foxit Reader.

**FRTaskPaneViewGetName**

It is called by Foxit Reader to receive the name of task pane view you want to create.

**FRTaskPaneViewGetTitle**

It is called by Foxit Reader to receive the title of task pane view you want to create.

**FRTaskPaneViewGetDropMenuItemText**

It is called by Foxit Reader to receive the text of drop-down menu at the right-up corner.

**FRTaskPaneViewGetDropMenuItemTip**

It is called by Foxit Reader to receive the tooltip of drop-down menu at the right-up corner.

**FRTaskPaneViewCreateHwnd**

It is called by Foxit Reader to receive the window handle of task pane view. Once the document is opened in browser, this callback will be called to create a window of task pane view.

**FRTTaskPaneViewDestroyHwnd**

It is called by Foxit Reader to destroy the window handle of task pane view. Once the document is closed in browser, this callback will be called to destroy the window of task pane view.

**FRTTaskPaneViewGetBigIcon**

This interface is reserved.

**FRTTaskPaneViewGetSmallIcon**

This interface is reserved.

**FRTTaskPaneViewOnActive**

It is called when the task pane view is to be activated.

**FRTTaskPaneViewOnDestroy**

It is called when the task pane view is to be destroyed.

**FRTTaskPaneViewOnShowTaskPane**

It is called when the task pane view is to be shown or to be hidden.

**FRTTaskPaneViewCanClose**

It is called when the task pane view is to be closed.

**FRTTaskPanelViewIsShowInMenu**

The callback is invoked by Foxit Reader to dispatch the message to plug-in.

**FRWndProviderCreateViewWnd**

It is called by Foxit Reader to notify the plug-in to create the view.

**FRWndProviderOnHScroll**

It is called by Foxit Reader when the user clicks a horizontally-aligned scroll bar, spin button control, or slider control.

**FRWndProviderOnVScroll**

It is called by Foxit Reader when the user clicks a vertically-aligned scroll bar, spin button control, or slider control.

**FRWndProviderOnCmdMsg**

It is called by the *Foxit Reader* to route and dispatch command messages and to handle the update of command user-interface objects, such as menu, toolbar.

**FRWndProviderMoveWindow**

It is called by the *Foxit Reader* to notify the plug-in to move the window.

**FRWndProviderOnSetCursor**

It is called by the *Foxit Reader* to notify the plug-in to set the cursor. References to *MFC CWnd::OnSetCursor* for detailed description.

**FRWndProviderZoomToPage**

It is called by the *Foxit Reader* to notify the plug-in to zoom the window.

**FRWndProviderGotoPage**

It is called by the *Foxit Reader* to notify the plug-in to go to the specified page.

**FRWndProviderShowWindow**

It is called by the *Foxit Reader* to notify the plug-in to show the window created.

**FRWndProviderReInitScrollBar**

This interface is reserved.

**FRWndProviderGetPageIndex**

This interface is reserved.

**FRWndProviderInitScrollBar**

It is called by the *Foxit Reader* to notify the plug-in to init the scroll bar.

**FRWndProviderOnSetFocus**

It is called by the *Foxit Reader* after the window gains the input focus.

**FRWndProviderOnMouseWheel**

It is called by the *Foxit Reader* as a user rotates the mouse wheel and encounters the wheel's next notch.

#### [FRWndProviderOnRelease](#)

It is called by the *Foxit Reader* to notify the plug-in to release the resource.

### Callbacks detail

#### FRBulbMsgCheckCallback

##### Syntax

```
typedef void (*FRBulbMsgCheckCallback)(  
    FS_BOOL bCheck,  
    void* clientData  
)
```

##### Description

The callback is invoked by Foxit Reader when user clicks the check box of the bulb message. The check box is used to check that whether to not show the bulb message again.

##### Parameter

---

bCheck	[In] Indicates that whether to not show the bulb message again.
--------	---

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

##### Return

void.

##### Head file reference

fr\_barExpT.h: 1557

##### Related method

[FRBulbMsgCenterAddMessage](#)

##### Since

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

#### FRBulbMsgOperationCallback

##### Syntax

```
typedef void (*FRBulbMsgOperationCallback)(  
    void clientData  
)
```

##### Description

The callback is invoked by Foxit Reader when user clicks the operation button of the bulb message.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void.

**Head file reference**

fr\_barExpT.h: 1543

**Related method**[FRBulbMsgCenterAddMessage](#)**Since**[SDK LATEEST VERSION > 7.3.1](#)**FRComputeEnabledProc****Syntax**

```
typedef FS_BOOL (*FRComputeEnabledProc)(
    void clientData
);
```

**Description**

A callback that is used to determine whether a menu item, toolbar button is enabled. This procedure is called every time the menu or toolbar button is displayed, so it should not do compute-time-intensive processing. It is called before the menu item or toolbar button is displayed. If it returns [FALSE](#), the menu item, or toolbar button disabled; otherwise it is enabled. If this callback is [NULL](#), the menu item or toolbar button is always enabled.

**Parameter**


---

clientData	[In] User-supplied data that was passed when <a href="#">FRToolButtonSetClientData</a> () or <a href="#">FRMenuItemSetClientData</a> () was called.
------------	---

---

**Return**[TRUE](#) if the menu item, or toolbar button is enabled, [FALSE](#) otherwise.**Head file reference**

fr\_menuExpT.h: 157

**Note:** Each menu item, toolbar button can have its own FRComputeEnabledProc(), or they can be shared.

**FRComputeMarkedProc****Syntax**

```
typedef FS_BOOL (*FRComputeMarkedProc)(  
    void clientData  
);
```

### Description

A callback that is used to determine whether a menu item or toolbar button is marked (a marked menuitem has a check mark next to it, and a marked toolbar button appears selected). It is called before the menu item or toolbar button is displayed. If it returns [FALSE](#), the menuitem of toolbar button is not marked; otherwise it is marked.

### Parameter

---

clientData	[In] User-supplied data that was passed when <a href="#">FRToolButtonSetClientData</a> () or <a href="#">FRMenuItemSetClientData</a> () was called.
------------	---

---

### Return

[TRUE](#) if the menu item, or toolbar button is enabled, [FALSE](#) otherwise.

### Head file reference

fr\_menuExpT.h: 174

**Note:** Each menu item, toolbar button can have its own FRComputeMarkedProc(), or they can be shared.

## FRDropDownProc

### Syntax

```
typedef void (*FRDropDownProc)(  
    void clientData  
);
```

### Description

The callback is invoked by Foxit Reader when user clicks the ribbon element drop-down arrow

### Parameter

---

clientData	[In] User-supplied data.
------------	--------------------------

---

### Return

void

### Head file reference

fr\_menuExpT.h: 197

## FRExecuteProc

**Syntax**

```
typedef void (*FRExecuteProc)(  
    void clientData  
)
```

**Description**

A callback that is called whenever a menu item, toolbar button or ribbon element is executed. It implements whatever themenu item, toolbar button or ribbon element does.

**Parameter**

---

clientData	[In] User-supplied data that was passed when <a href="#">FRToolButtonSetClientData</a> , <a href="#">FRMenuItemSetClientData</a> or <a href="#">FRRibbonBarAddButtonToAddPlace</a> was called.
------------	--

---

**Return**

void

**Head file reference**

fr\_menuExpT.h: 139

**FRFreeDataProc****Syntax**

```
typedef void (*FRFreeDataProc)(  
    void clientData  
)
```

**Description**

A callback that is used to free data. It is called when Foxit Reader will free objects such as ribbon element.

**Parameter**

---

clientData	[In] User-supplied data.
------------	--------------------------

---

**Return**

void

**Head file reference**

fr\_menuExpT.h: 187

**FRGetMessageStringProc****Syntax**

```
typedef FS_BOOL (*FRGetMessageStringProc)(  
    FS_UINT nID,  
    FS_WideString outMessage
```

);

**Description**

A callback that is called whenever a menu is popped up to receive the tooltip.

**Parameter**

nID	[In] The specified ID of the pop-up menu item.
outMessage	[Out] It receives the message.

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_menuExpT.h: 209

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2](#)

**FRPopUpMenuProc****Syntax**

```
typedef void (*FRPopUpMenuProc)(  
    void pClientData,  
    FR\_Menu popUpMenu  
)
```

**Description**

A callback that is called when the context menu is popped up in the doc view.

**Parameter**

pClientData	[In] User-supplied data that was passed when <a href="#">FRToolButtonSetClientData</a> () or <a href="#">FRMenuItemSetClientData</a> () was called.
popUpMenu	[In] The menu popped up.

**Return**

void

**Head file reference**

fr\_appExpT.h: 3693

**FRRibbonElementImageInitProc****Syntax**

```
typedef void (*FRRibbonElementImageInitProc)(  
    FR\_RibbonElement ribbonElement,  
    void* clientData  
)
```

**Description**

The callback is invoked by Foxit Reader when the ribbon object is to init the image.

**Parameter**

ribbonElement	[In] The ribbon element is to init the image.
---------------	---

clientData	[In] The client data associated with the ribbon element.
------------	--

**Return**

void

**Head file reference**

[fr\\_barExpT.h: 1406](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRRibbonPaletteButtonHighlightItemProc****Syntax**

```
typedef void (*FRRibbonPaletteButtonHighlightItemProc)(  
    void* pClientData,  
    FS\_INT32 nIndex,  
    FS\_BOOL bHighlight  
)
```

**Description**

The callback is invoked by Foxit Reader when the mouse is over ribbon palette object.

**Parameter**

pClientData	[In] The client data associated with the ribbon palette object.
-------------	---

nIndex	[In] The index of the item.
--------	-----------------------------

bHighlight	[In] Whether the item is highlight or not.
------------	--

**Return**

void

**Head file reference**

fr\_barExpT.h: 1433

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

**FRRibbonPanelImageInitProc****Syntax**

```
typedef void (*FRRibbonPanelImageInitProc)(  
    FR.RibbonPanel ribbonPanel,  
    void* reserved  
)
```

**Description**

The callback is invoked by Foxit Reader when the ribbon object is to init the image.

**Parameter**

---

ribbonPanel	[In] The ribbon panel is to init the image.
-------------	---

---

reserved	[In] It is reserved.
----------	----------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1419

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRActionHandlerDocOpen****Syntax**

```
typedef FS_BOOL (*FRActionHandlerDocOpen)(  
    FS\_LPVOID clientData,  
    FPD\_Action action,  
    FR.Document frDoc,  
    FR.DocView frDocView,  
    FS\_BOOL bDisableGoto  
)
```

**Description**

It is called by Foxit Reader if the document contains an action to be performed when the document is opened. If you do not want to customize the action process, you can set this callback to NULL or you must return FALSE. You can call [FPDACTIONGETTYPE](#) to get the type of the action. You can refer to [FPD\\_ACTIONTYPE](#). About theopen action, you can refer to *PDF Reference, Section3.6, Entries In the Catalog Dictionary*.

### Parameter

---

clientData	[In] The user-supplied data.
action	[In] The action to be performed when the document is opened.
frDoc	[In] The currently opened document.
frDocView	[In] The current document view.
bDisableGoto	[In] It indicates whether the plug-in has to disable the Go-To Actions.

---

### Return

TRUE if the plug-in customizes the action process successfully. If you do not want to customize the action process, you must return FALSE, so that Foxit Reader can implement the default action process.

### Head file reference

fr\_appExpT.h: 5315

### Group

[FR\\_ActionHandlerCallbacksRec](#)

### Since

[SDK LATEEST VERSION > 2.1](#)

## FRACTIONHANDLERJAVASCRIPT

### Syntax

```
typedef FS_BOOL (*FRACTIONHANDLERJAVASCRIPT)(  
    FS_LPVVOID clientData,  
    FPD_Action action,  
    FS_LPCWSTR lpwsJSName,  
    FR_Document frDoc,  
    FR_DocView frDocView  
)
```

### Description

It is called by Foxit Reader if the document contains a JavaScript action to be performed when the document is opened. If you do not want to customize the action process, you can set this callback to NULL or you must return FALSE. You can call [FPDACTIONGETTYPE](#) to

get the type of the action. You can refer to [FPD\\_ActionType](#). About the JavaScriptaction, you can refer to *PDF Reference, Section8.6, JavaScript Actions*.

### Parameter

---

clientData	[In] The user-supplied data.
action	[In] The JavaScript action to be performed when the document is opened.
lpwsJSName	[In] The name of the JavaScript.
frDoc	[In] The currently opened document.
frDocView	[In] The current document view.

---

### Return

TRUE if the plug-in customizes the action process successfully. If you do not want to customize the action process, you must returnFALSE, so that Foxit Reader can implement the default action process.

### Head file reference

fr\_appExpT.h: 5335

### Group

[FR\\_ActionHandlerCallbacksRec](#)

### Since

[SDK\\_LATEEST\\_VERSION > 2.1](#)

## FRACTIONHandlerPage

### Syntax

```
typedef FS_BOOL (*FRACTIONHandlerPage)(  
    FS_LPVOID clientData,  
    FPD_Action action,  
    FPD_AActionType type,  
    FR_Document frDoc,  
    FR_DocView frDocView  
>;
```

### Description

It is called by Foxit Reader if the page contains an additional-actions dictionary defining actions to be performedwhen the page is opened or closed. If you do not want to customize the action process, you can set this callback to NULL or you must return FALSE. Youcan call [FPDACTIONGetType](#) to get the type of the action. You can refer to [FPD\\_ActionType](#). About the additionalaction, you can refer to *PDF Reference, Section8.5, Trigger Events*.

**Parameter**


---

clientData	[In] The user-supplied data.
action	[In] The action to be performed when the page is opened or closed.
type	[In] The additional action type.
frDoc	[In] The currently opened document.
frDocView	[In] The current document view.

---

**Return**

TRUE if the plug-in customizes the action process successfully. If you do not want to customize the action process, you must return FALSE, so that Foxit Reader can implement the default action process.

**Head file reference**

fr\_appExpT.h: 5356

**Group**

[FR\\_ActionHandlerCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.1](#)

**FRACTIONHandlerLink****Syntax**

```
typedef FS_BOOL (*FRACTIONHandlerLink)(
    FS_LPVVOID clientData,
    FPD_Action action,
    FR_Document frDoc,
    FR_DocView frDocView
);
```

**Description**

It is called by Foxit Reader if the link annotation contains an actions to be performed when the annotation is activated. If you do not want to customize the action process, you can set this callback to NULL or you must return FALSE. You can call [FPDACTIONGetType](#) to get the type of the action. You can refer to [FPD\\_ActionType](#). About the linkannotation, you can refer to *PDF Reference, Section8.4, Link Annotations*.

**Parameter**

---

clientData	[In] The user-supplied data.
action	[In] The action to be performed when the annotation is activated.
frDoc	[In] The currently opened document.
frDocView	[In] The current document view.

---

**Return**

TRUE if the plug-in customizes the action process successfully. If you do not want to customize the action process, you must return FALSE, so that Foxit Reader can implement the default action process.

**Head file reference**

fr\_appExpT.h: 5375

**Group**

[FR\\_ActionHandlerCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.1](#)

**FRAppOnLangUIChange****Syntax**

```
typedef void (*FRAppOnLangUIChange)(  
    FS_LPVVOID clientData  
)
```

**Description**

A callback for application level event handler. It is called whenever the user change the language.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 728

**Group**

[FR\\_AppEventCallbacksRec](#)

## FRAppOnActivate

### Syntax

```
typedef void (*FRAppOnActivate)(  
    FS_LPVVOID clientData,  
    FS_BOOL bActive  
>);
```

### Description

A callback for application level event handler. It is called whenever the application is to being activated or deactivated.

### Parameter

clientData	[In] The user-supplied data.
bActive	[In] Specifies whether the application is being activated or deactivated. <a href="#">TRUE</a> means the application is being activated. <a href="#">FALSE</a> means the application is being deactivated.

### Return

#### Head file reference

fr\_appExpT.h: 742

#### Group

[FR\\_AppEventCallbacksRec](#)

## FRAppWillQuit

### Syntax

```
typedef void (*FRAppWillQuit)(  
    FS_LPVVOID clientData  
>);
```

### Description

A callback for application level event handler. It is called at the shutdown time.

### Parameter

clientData	[In] The user-supplied data.
------------	------------------------------

### Return

#### Head file reference

fr\_appExpT.h: 754

#### Group

[FR\\_AppEventCallbacksRec](#)**FRAppWillQuit****Syntax**

```
typedef void (*FRAppWillQuit)(  
    FS_LPVVOID clientData,  
    FS_LPCSTR lpModuleName  
>);
```

**Description**

A callback for application level event handler. It is called when Foxit Reader needs to download the updated module.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

lpModuleName	[In] The module name.
--------------	-----------------------

---

**Return****Head file reference**

fr\_appExpT.h: 767

**Group**[FR\\_AppEventCallbacksRec](#)**FRAppOnToolChanged****Syntax**

```
typedef void (*FRAppOnToolChanged)(  
    FS_LPVVOID clientData,  
    FR_Tool deactivateTool,  
    FR_Tool activeTool  
>);
```

**Description**

A callback for application level event handler. It is called when current active tool has been changed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

deactivateTool	[In] The deactivate tool.
----------------	---------------------------

---

---

activeTool	[In] The current active tool.
------------	-------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 781

**Group**[FR\\_AppEventCallbacksRec](#)**FRApOnSkinChange****Syntax**

```
typedef void (*FRApOnSkinChange)(  
    FS\_LPVOID clientData  
)
```

**Description**

A callback for application level event handler. It is called when Foxit Reader changed the UI skin.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 793

**Group**[FR\\_AppEventCallbacksRec](#)**FRApOnShowFullScreen****Syntax**

```
typedef void (*FRApOnShowFullScreen)(  
    FS\_LPVOID clientData  
)
```

**Description**

A callback for application level event handler. It is called when Foxit Reader starts to show fullscreen mode.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 805

**Group**[FR\\_AppEventCallbacksRec](#)**FRApOnHideFullScreen****Syntax**

```
typedef void (*FRApOnHideFullScreen)(  
    FS_LPVVOID clientData  
)
```

**Description**

A callback for application level event handler. It is called when Foxit Reader ends the fullscreen mode.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 817

**Group**[FR\\_AppEventCallbacksRec](#)**FRApOnResetToolbars****Syntax**

```
typedef void (*FRApOnResetToolbars)(  
    FS_LPVVOID clientData  
)
```

**Description**

A callback for application level event handler. It is called when Foxit Reader re-layout all toolbars.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 829

**Group**[FR\\_AppEventCallbacksRec](#)**FRAppOnRunCommandLine****Syntax**

```
typedef void (*FRAppOnRunCommandLine)(  
    FS\_LPVOID clientData  
);
```

**Description**

A callback for application level event handler. It is called when Foxit Reader startups in command-line mode.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 842

**Group**[FR\\_AppEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FRAppOnDrop****Syntax**

```
typedef void (*FRAppOnDrop)(  
    FS\_LPVOID clientData,  
    FS\_LPCWSTR lpszPath  
);
```

**Description**

A callback for application level event handler. It is called when the user drag a file to Foxit Reader and drop.

**Parameter**

---

clientData	[In] The user-supplied data.
lpszPath	[In] The path of the file that is dragged.

---

**Return****Head file reference**

fr\_appExpT.h: 856

**Group**[FR\\_AppEventCallbacksRec](#)**Since**[SDK LATEEST VERSION > 1.0](#)**FRApOnTextViewChange****Syntax**

```
typedef void (*FRApOnTextViewChange)(  
    FS_LPVVOID clientData,  
    FS_INT32 nViewType  
)
```

**Description**

A callback for application level event handler. It is called when Foxit Reader changes view mode.

**Parameter**


---

clientData	[In] The user-supplied data.
nViewType	[In] The current display mode: 0 for preview 1 for text.

---

**Return****Head file reference**

fr\_appExpT.h: 870

**Group**[FR\\_AppEventCallbacksRec](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FRApOnRecentFileListChange**

**Syntax**

```
typedef void (*FRAppOnRecentFileListChange)(  
    FS_LPVOID clientData,  
    FS_WideStringArray arrFileList  
) ;
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when the recent file list changes.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

arrFileList	[In] The recent file list.
-------------	----------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 884

**Group**

[FR\\_AppEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRAppOnShowRibbonFilePage****Syntax**

```
typedef void (*FRAppOnShowRibbonFilePage)(  
    FS_LPVOID clientData  
) ;
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when the ribbon file page shows.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 897

**Group**

[FR\\_AppEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRApOnCloseRibbonFilePage****Syntax**

```
typedef void (*FRApOnCloseRibbonFilePage)(  
    FS_LPVVOID clientData  
)
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when the ribbon file page is closed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 910

**Group**[FR\\_AppEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRApOnRibbonElementsLoadFinish****Syntax**

```
typedef void (*FRApOnRibbonElementsLoadFinish)(  
    FS_LPVVOID clientData,  
    void* pParentWnd  
)
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when the ribbon elements finish loading.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

pParentWnd	[In] A pointer to the parent window. It represents the <i>MFC CWnd*</i> .
------------	---

---

**Return****Head file reference**

fr\_appExpT.h: 924

**Group**[FR\\_AppEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRApOnOpenDocument****Syntax**

```
typedef FS_BOOL (*FRApOnOpenDocument)(  
    FS_LPOID clientData,  
    FS_LPCWSTR lpszFilePath  
)
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when Foxit Reader is going to open a document. Returning TRUE means the plug-in will take over the process, and the Foxit Reader will terminate the process, otherwise the Foxit Reader will continue the opening process.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

lpszFilePath	[In] The file path to be opened.
--------------	----------------------------------

---

**Return**

TRUE means the plug-in will take over the process, and the Foxit Reader will terminate the process, otherwise the Foxit Reader will continue the opening process.

**Head file reference**

fr\_appExpT.h: 941

**Group**[FR\\_AppEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)**FRApOnRibbonCategoryClicked**

**Syntax**

```
typedef void (*FRAppOnRibbonCategoryClicked)(  
    FS\_LPVVOID clientData,  
    FS\_BOOL bChanged  
)
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when the ribbon category is clicked.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

bChanged	[In] Whether the ribbon category is changed or not.
----------	---

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 956

**Group**

[FR\\_AppEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRAppOnMainFrmLoadFinish****Syntax**

```
typedef void (*FRAppOnMainFrmLoadFinish)(  
    FS\_LPVVOID clientData  
)
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when the main frame finishes loading.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 969

**Group**[FR\\_AppEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 3.0](#)**FRAppOnRibbonUILayoutFinish****Syntax**

```
typedef void (*FRAppOnRibbonUILayoutFinish)(  
    FS\_LPVOID clientData,  
    void* pParentWnd  
);
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when the ribbon elements finish laying out.

**Parameter**

---

clientData	[In] The user-supplied data.
pParentWnd	[In] A pointer to the parent window. It represents the <i>MFC CWnd*</i> .

---

**Return****Head file reference**

fr\_appExpT.h: 983

**Group**[FR\\_AppEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 3.0.0.0](#)**FRAppOnShowTaskPane****Syntax**

```
typedef void (*FRAppOnShowTaskPane)(  
    FS\_LPVOID clientData,  
    FS\_BOOL bShow  
);
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when the task pane is being shown.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

bShow	[In] Whether the task pane is to be shown or not.
-------	---

---

**Return****Head file reference**

fr\_appExpT.h: 997

**Group**[FR\\_AppEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 3.0.0.0](#)**FRApOnShowPopupMenu****Syntax**

```
typedef void (*FRApOnShowPopupMenu)(  
    FS_LPVOID clientData  
)
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when the pop-up menu is shown.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 1010

**Group**[FR\\_AppEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)**FRApOnDidOpenAllFiles****Syntax**

```
typedef void (*FRAppOnDidOpenAllFiles)(  
    FS\_LPVOID clientData  
);
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when all the files have been opened.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 1023

**Group**

[FR\\_AppEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

**FRAppOnMainFrameWillClose****Syntax**

```
typedef void (*FRAppOnMainFrameWillClose)(  
    FS\_LPVOID clientData,  
    FS\_BOOL* bCanClose  
);
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when the main frame is to be closed. Then the plug-in can determine whether the main frame can be closed or not.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

bCanClose	[Out] Determines whether the main frame can be closed or not.
-----------	---

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 1038

**Group**[FR\\_AppEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)**FRAppMainFrameOnSize****Syntax**

```
typedef void (*FRAppMainFrameOnSize)(  
    FS\_LPVOID clientData  
);
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when the main frame's size is changing.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 1052

**Group**[FR\\_AppEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)**FRAppMainFrameOnLoadWinPlacementFinish****Syntax**

```
typedef void (*FRAppMainFrameOnLoadWinPlacementFinish)(  
    FS\_LPVOID clientData,  
    HWND hMainframe  
);
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when the size and location of a window from the registry is loaded.

**Parameter**

---

clientData	[In] The user-supplied data.
hMainframe	[In] A handle of the frame window.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 1067

**Group**[FR\\_AppEventCallbacksRec](#)**Since**[SDK LATEST VERSION > 7.3.1](#)**FRApplianceOnLoaded****Syntax**

```
typedef void (*FRApplianceOnLoaded)(  
    FS\_LPVOID clientData,  
    FS\_LPCWSTR lpszPluginName  
)
```

**Description**

A callback for application level event handler. It is called by Foxit Reader when the plugin is loaded.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---



---

lpszPluginName	[In] the plugin name.
----------------	-----------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 1082

**Group**[FR\\_AppEventCallbacksRec](#)**Since**[SDK LATEST VERSION > 7.3.1](#)

## FRAppOnDidCloseRibbonFilePage

### Syntax

```
typedef void (*FRAppOnDidCloseRibbonFilePage)(  
    FS_LPVVOID clientData  
)
```

### Description

A callback for application level event handler. It is called by Foxit Reader when the ribbon file page had closed.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

### Return

#### Head file reference

fr\_appExpT.h: 1095

#### Group

[FR\\_AppEventCallbacksRec](#)

#### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRCaptureGetType

### Syntax

```
typedef FS_LPSTR (*FRCaptureGetType)(  
    FS_LPVVOID clientData  
)
```

### Description

It is called to get the capture type. See the group definition [CaptureType](#).

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

### Return

The capture type

#### Head file reference

fr\_appExpT.h: 4064

#### Group

[FR\\_CaptureCallbacksRec](#)

## FRCaptureLosingCapture

### Syntax

```
typedef void (*FRCaptureLosingCapture)(  
    FS_LPVVOID clientData,  
    FR_Document document,  
    void* curCaptureData  
)
```

### Description

It is called by Foxit Reader to notify that the current capture handler is losing capture.

### Parameter

clientData	[In] The user-supplied data.
document	[In] The input document whose capture handler is losing capture.
curCaptureData	[In] The current capture data for doc.

### Return

#### Head file reference

fr\_appExpT.h: 4076

#### Group

[FR\\_CaptureCallbacksRec](#)

## FRCaptureGettingCapture

### Syntax

```
typedef void (*FRCaptureGettingCapture)(  
    FS_LPVVOID clientData,  
    FR_Document document,  
    void* curCaptureData  
)
```

### Description

It is called by Foxit Reader to notify that the current capture handler is getting capture.

### Parameter

clientData	[In] The user-supplied data.
------------	------------------------------

document	[In] The input document whose capture handler is getting capture.
----------	---

---

curCaptureData	[In] The current capture data for doc.
----------------	--

---

**Return****Head file reference**

fr\_appExpT.h: 4088

**Group**[FR\\_CaptureCallbacksRec](#)**FRCaptureLButtonDown****Syntax**

```
typedef FS_BOOL (*FRCaptureLButtonDown)(  
    FS_LVOID clientData,  
    FR_PageView pageView,  
    void* curCaptureData,  
    unsigned int nFlags,  
    FS_DevicePoint point  
)
```

**Description**

It is called by Foxit Reader to notify the left-button-down event.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

---

pageView	[In] The input page view whose left-button-down event occurred.
----------	---

---

curCaptureData	[In] The current capture data for doc.
----------------	--

---

nFlags	[In] Indicates whether various virtual keys are down.
--------	---

---

point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.
-------	---

---

**Return**

[TRUE](#) if the left-button-down event was handled, otherwise not.

**Head file reference**

---

fr\_appExpT.h: 4102

**Group**[FR\\_CaptureCallbacksRec](#)**FRCaptureLButtonUp****Syntax**

```
typedef FS_BOOL (*FRCaptureLButtonUp)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    void* curCaptureData,  
    unsigned int nFlags,  
    FS_DevicePoint point  
)
```

**Description**

It is called by Foxit Reader to notify the left-button-up event.

**Parameter**


---

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose left-button-up event occurred.
curCaptureData	[In] The current capture data for doc.
nFlags	[In] Indicates whether various virtual keys are down.
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

---

**Return**

[TRUE](#) if the left-button-up event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 4116

**Group**[FR\\_CaptureCallbacksRec](#)**FRCaptureLButtonDblClk****Syntax**

```
typedef FS_BOOL (*FRCaptureLButtonDblClk)(
```

```
FS_LPVVOID clientData,  
FR_PageView pageView,  
void* curCaptureData,  
unsigned int nFlags,  
FS_DevicePoint point  
);
```

**Description**

It is called by Foxit Reader to notify the left-button-DblClk event.

**Parameter**

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose left-button-DblClk event occurred.
curCaptureData	[In] The current capture data for doc.
nFlags	[In] Indicates whether various virtual keys are down.
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

**Return**

[TRUE](#) if the left-button-DblClk event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 4130

**Group**

[FR\\_CaptureCallbacksRec](#)

**FRCaptureMouseMove****Syntax**

```
typedef FS_BOOL (*FRCaptureMouseMove)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    void* curCaptureData,  
    unsigned int nFlags,  
    FS_DevicePoint point  
)
```

**Description**

It is called by Foxit Reader to notify the mouse-move event.

**Parameter**


---

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose mouse-move event occurred.
curCaptureData	[In] The current capture data for doc.
nFlags	[In] Indicates whether various virtual keys are down.
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

---

**Return**

[TRUE](#) if the left-button-DblClk event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 4144

**Group**

[FR\\_CaptureCallbacksRec](#)

**FRCaptureRButtonDown****Syntax**

```
typedef FS_BOOL (*FRCaptureRButtonDown)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    void* curCaptureData,  
    unsigned int nFlags,  
    FS_DevicePoint point  
)
```

**Description**

It is called by Foxit Reader to notify the right-button-down event.

**Parameter**


---

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose right-button-down event occurred.

---

---

curCaptureData	[In] The current capture data for doc.
nFlags	[In] Indicates whether various virtual keys are down.
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

---

**Return**

[TRUE](#) if the left-button-DblClk event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 4158

**Group**

[FR\\_CaptureCallbacksRec](#)

**FRCaptureRButtonUp****Syntax**

```
typedef FS_BOOL (*FRCaptureRButtonUp)(
    FS_LPVVOID clientData,
    FR_PageView pageView,
    void* curCaptureData,
    unsigned int nFlags,
    FS_DevicePoint point
);
```

**Description**

It is called by Foxit Reader to notify the right-button-up event.

**Parameter**


---

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose right-button-up event occurred.
curCaptureData	[In] The current capture data for doc.
nFlags	[In] Indicates whether various virtual keys are down.
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

---

**Return**

[TRUE](#) if the left-button-DblClk event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 4172

**Group**

[FR\\_CaptureCallbacksRec](#)

**FRCaptureRButtonDblClk****Syntax**

```
typedef FS_BOOL (*FRCaptureRButtonDblClk)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    void* curCaptureData,  
    unsigned int nFlags,  
    FS_DevicePoint point  
) ;
```

**Description**

It is called by Foxit Reader to notify the right-button-DblClk event.

**Parameter**

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose right-button-DblClk event occurred.
curCaptureData	[In] The current capture data for doc.
nFlags	[In] Indicates whether various virtual keys are down.
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

**Return**

[TRUE](#) if the left-button-DblClk event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 4186

**Group**

[FR\\_CaptureCallbacksRec](#)

## FRCaptureContextMenu

### Syntax

```
typedef FS_BOOL (*FRCaptureContextMenu)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    FS_DevicePoint point  
)
```

### Description

It is called by Foxit Reader to notify the context menu pop-up event.

### Parameter

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose context menu pop-up event occurred.
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

### Return

[TRUE](#) if the context menu pop-up event was handled, otherwise not.

### Head file reference

fr\_appExpT.h: 4199

### Group

[FR\\_CaptureCallbacksRec](#)

### Since

[SDK LATEEST VERSION > 8.3.2](#)

## FRConProviderOnFileOpen

### Syntax

```
typedef FS_BOOL (*FRConProviderOnFileOpen)(  
    FS_LPVVOID clientData,  
    FR_Document doc,  
    FS_LPCWSTR lpszSource,  
    FS_BOOL bIsAttachment  
)
```

### Description

It will be invoked by Foxit Reader when a document is opened. If it returns [TRUE](#), it means the plug-in can process the document and the other callbacks go on. Otherwise not.

#### Parameter

---

clientData	[In] The user-supplied data.
doc	[In] It identifies the document opened. It can be used as a unique identifier for the document.
lpszSource	[In] The full path of the document opened.
bIsAttachment	[In] TRUE means the document is a attachment.

---

#### Return

[TRUE](#) means the plug-in can process this document and the other callbacks go on. Otherwise not.

#### Head file reference

fr\_appExpT.h: 2811

#### Group

[FR\\_ContentProviderCallbacksRec](#)

### FRCOnProviderOnGetPermissions

#### Syntax

```
typedef unsigned long (*FRCOnProviderOnGetPermissions)(  
    FS\_LPVVOID clientData,  
    FR\_Document doc,  
    unsigned long pdfselfPermissions  
>;
```

#### Description

The plug-in can control the permissions of the document.

#### Parameter

---

clientData	[In] The user-supplied data.
doc	[In] It identifies the document opened. It can be used as a unique identifier for the document.
pdfselfPermissions	[In] The PDF document's permission.

---

**Return**

The user-defined permissions.

**Head file reference**

fr\_appExpT.h: 2823

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**FRConProviderOnGetContentSize****Syntax**

```
typedef FS_BOOL (*FRConProviderOnGetContentSize)(  
    FS_LVOID clientData,  
    FR_Document doc,  
    unsigned long* pTotalSize  
>);
```

**Description**

It will be invoked by Foxit Reader to receive the size of the document the plug-in provides.

**Parameter**

clientData	[In] The user-supplied data.
doc	[In] It identifies the document opened. It can be used as a unique identifier for the document.
pTotalSize	[Out] It receives the size of the document the plug-in provides.

**Return**

[TRUE](#) means successful. Otherwise not.

**Head file reference**

fr\_appExpT.h: 2835

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**FRConProviderOnReadContent****Syntax**

```
typedef FS_BOOL (*FRConProviderOnReadContent)(  
    FS_LVOID clientData,  
    FR_Document doc,  
    FS_DWORD pos,  
    unsigned char* pBuf,  
    unsigned long size
```

---

);

**Description**

It will be invoked by Foxit Reader to receive the document data the plug-in provides. This routine occurs times without number and the data is read in blocks.

**Parameter**

clientData	[In] The user-supplied data.
doc	[In] It identifies the document opened. It can be used as a unique identifier for the document.
pos	[In] It identifies the position of the document data the plug-in provides.
pBuf	[Out] It receives the data block the plug-in provides every time.
size	[In] It identifies the size of the data block that the plug-in should provide every time.

**Return**

TRUE means successful. Otherwise not.

**Head file reference**

fr\_appExpT.h: 2850

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**FRCOnProviderOnWriteContent****Syntax**

```
typedef FS_BOOL (*FRCOnProviderOnWriteContent)(  
    FS_LPVVOID clientData,  
    FR_Document doc,  
    unsigned char* pBuf,  
    unsigned long size,  
    FS_LPCWSTR lpSaveFilePath  
)
```

**Description**

It will be invoked by Foxit Reader when the document is modified and is saved. The plug-in must re-encrypt the document and save the document in some format.

**Parameter**

---

clientData	[In] The user-supplied data.
doc	[In] It identifies the document opened. It can be used as a unique identifier for the document.
pBuf	[In] Foxit Reader passes the document data that is modified to the plug-in.
size	[In] The buffer size.
lpSaveFilePath	[In] Foxit Reader passes the file path where the document is saved to the plug-in.

---

**Return**

[TRUE](#) means successful. Otherwise not.

**Head file reference**

fr\_appExpT.h: 2865

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**FRConProviderOnFileClose****Syntax**

```
typedef void (*FRConProviderOnFileClose)(  
    FS\_LPVOID clientData,  
    FR\_Document doc  
)
```

**Description**

It will be invoked by Foxit Reader when the document is closed.

**Parameter**


---

clientData	[In] The user-supplied data.
doc	[In] It identifies the document opened. It can be used as a unique identifier for the document.

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 2876

**Group**[FR\\_ContentProviderCallbacksRec](#)**FRConProviderOnWriteAttachmentContent****Syntax**

```
typedef FS_BOOL (*FRConProviderOnWriteAttachmentContent)(
    FS\_LPVVOID clientData,
    FR\_Document frDoc,
    unsigned char * pBuf,
    unsigned long size,
    FS\_LPCWSTR lpAttachmntPath
);
```

**Description**

When the attachment is opened, it will be saved to a temp directory first. This interface leaves the choice of protecting the attachment to the plug-in. You can get the type of the attachment from the file path.

**Parameter**

clientData	[In] The user-supplied data.
frDoc	[In] It is the document whose attachment is opened.
pBuf	[In] The content of the attachment.
size	[In] The size of the attachment.
lpAttachmntPath	[In] The path of the attachment.

**Return**

[TRUE](#) means successful, otherwise failure.

**Head file reference**

fr\_appExpT.h: 2891

**Group**[FR\\_ContentProviderCallbacksRec](#)**FRConProviderOnGetAttachmentSize****Syntax**

```
typedef FS_BOOL (*FRConProviderOnGetAttachmentSize)(
    FS\_LPVVOID clientData,
```

---

```
FR_Document frDoc,
unsigned long* pTotalSize,
FS_LPCWSTR lpAttachmntPath
);
```

**Description**

When the attachment is modified, the Foxit Reader needs to reattach the attachment. This interface is invoked to get the total size of the attachment modified.

**Parameter**

clientData	[In] The user-supplied data.
frDoc	[In] It is the document whose attachment is opened.
pTotalSize	[Out] The total size of the attachment.
lpAttachmntPath	[In] The path of the attachment.

**Return**

TRUE means successful, otherwise failure.

**Head file reference**

fr\_appExpT.h: 2905

**Group**

FR\_ContentProviderCallbacksRec

**FRCOnProviderOnReadAttachmentContent****Syntax**

```
typedef FS_BOOL (*FRCOnProviderOnReadAttachmentContent)(
    FS_LPVOID clientData,
    FR_Document frDoc,
    unsigned char* pBuf,
    unsigned long size,
    FS_LPCWSTR lpAttachmntPath
);
```

**Description**

When the attachment is modified, the Foxit Reader needs to reattach the attachment. This interface is invoked to get the content of the attachment modified. Since SDK\_LATEST\_VERSION > 3.0, FRCOnProviderOnReadAttachmentContent is replaced with FRCOnProviderOnReadAttachmentContentInBlocks.

**Parameter**

---

clientData	[In] The user-supplied data.
frDoc	[In] It is the document whose attachment is opened.
pBuf	[Out] It receives the content of the attachment modified.
size	[In] The total size of the attachment.
lpAttchmntPath	[In] The path of the attachment.

---

**Return**

[TRUE](#) means successful, otherwise failure.

**Head file reference**

fr\_appExpT.h: 2921

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**FRCOnProviderIsPageAvail****Syntax**

```
typedef FS_BOOL (*FRCOnProviderIsPageAvail)(
    FS\_LPVVOID clientData,
    FR\_Document frDoc,
    FS\_INT32 iPage,
    clientData,
    frDoc,
    iPage,
    FS\_FLOAT* outWidth,
    FS\_FLOAT* outHeight,
    clientData,
    frDoc,
    iPage,
    FS\_DIBitmap bitmap,
    FS\_INT32 start_x,
    FS\_INT32 start_y,
    FS\_INT32 width,
    FS\_INT32 height
);
```

**Description**

It will be invoked by Foxit Reader when a PDF page data can not be obtained and the PDF page will be rendered. Must return [FALSE](#) if the PDF file is a local file and have all content information, and Foxit Reader will get page size by itself.

**Parameter**

clientData	[In] The user-supplied data.
frDoc	[In] It is the document whose attachment is opened.
iPage	[In] The page index specifies the PDF page to be check.
clientData	[In] The user-supplied data.
frDoc	[In] It is the document whose attachment is opened.
iPage	[In] The page index specifies the PDF page to get the page size.
outWidth	[Out] The page width of specified page by <i>iPage</i> in PDF coordinate, client must fill the page width to this parameter.
outHeight	[Out] The page height of specified page by <i>iPage</i> in PDF coordinate, client must fill the page height to this parameter.
clientData	[In] The user-supplied data.
frDoc	[In] It is the document whose attachment is opened.
iPage	[In] The page index specifies the PDF page to be rendered.
bitmap	[In] The bitmap passed by Foxit reader, client will be draw owner information to it.
start_x	[In] Left pixel position of the drawing area in the device coordination.
start_y	[In] Top pixel position of the display area in the device coordination.
width	[In] Horizontal size (in pixels) for drawing area.
height	[In] Vertical size (in pixels) for drawing area.

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_appExpT.h: 2968

**Group**[FR\\_ContentProviderCallbacksRec](#)

**Note:** Must be return FALSE if client needn't draw owner information to a page view. If return TRUE, Foxit Reader justcall this callback to drawing, and the content of PDF page will not be displayed.

**FRConProviderNeedReopenDoc****Syntax**

```
typedef FS_BOOL (*FRConProviderNeedReopenDoc)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc  
)
```

**Description****Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] It is the document whose attachment is opened.
-------	---

---

**Return**

[TRUE](#) for needing to reopen the document.

**Head file reference**

fr\_appExpT.h: 2978

**Group**[FR\\_ContentProviderCallbacksRec](#)**FRConProviderCanBeSaved****Syntax**

```
typedef FS_BOOL (*FRConProviderCanBeSaved)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc  
)
```

**Description****Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] It is the document whose attachment is opened.
-------	---

---

**Return**

[TRUE](#) for the document can be saved.

**Head file reference**

fr\_appExpT.h: 2988

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**FRContentProviderIsProcessErrMsg****Syntax**

```
typedef FS_BOOL (*FRContentProviderIsProcessErrMsg)(
    FS\_LPVVOID clientData
);
```

**Description**

Checks whether the plug-in processes the error message.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

whether the plug-in processes the error message.

**Head file reference**

fr\_appExpT.h: 2998

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**FRContentProviderIsSupportPanel****Syntax**

```
typedef FS_BOOL (*FRContentProviderIsSupportPanel)(
    FS\_LPVVOID clientData,
    FR\_Document frDoc
);
```

**Description**

Checks Whether the document supports panel.

**Parameter**

---

clientData	[In] The user-supplied data.
frDoc	[In] Whether the document supports panel.

---

**Return**

Whether the document supports panel.

**Head file reference**

fr\_appExpT.h: 3009

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**FRContentProviderIsSupportViewByScroll****Syntax**

```
typedef FS_BOOL (*FRContentProviderIsSupportViewByScroll)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc  
)
```

**Description**

Checks whether the document supports viewing by scrolling.

**Parameter**

---

clientData	[In] The user-supplied data.
frDoc	[In] Whether the document supports viewing by scrolling.

---

**Return**

Whether the document supports viewing by scrolling.

**Head file reference**

fr\_appExpT.h: 3020

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**FRContentProviderIsSupportViewByScroll****Syntax**

```
typedef FS_BOOL (*FRContentProviderIsSupportViewByScroll)(
```

---

```
FS_LPVOID clientData,
FS_LPCSTR lpsName
);
```

**Description**

Determine whether to delete the navigation panel.

**Parameter**


---

clientData	[In] The user-supplied data.
lpsName	[In] The name of navigation panel.

---

**Return**

Returns [TRUE](#) to delete the navigation panel, otherwise not.

**Head file reference**

fr\_appExpT.h: 3031

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**FROnGetPageDictObjectNumber****Syntax**

```
typedef FS_BOOL (*FROnGetPageDictObjectNumber)(
    FS_LPVOID clientData,
    FR_Document frDoc,
    FS_INT32 nPageIndex,
    FS_DWORD* outObjNumber
);
```

**Description**

This interface is reserved.

**Parameter**


---

clientData	[In] The user-supplied data.
frDoc	[In] The document.
nPageIndex	[In] The page index.
outObjNumber	[Out] It receives the object number of page dictionary.

---

**Return**

Nonzero if the plug-in gets the number; otherwise 0.

**Head file reference**

fr\_appExpT.h: 3044

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**FRConProviderIsWriteContentProgressive****Syntax**

```
typedef FS_BOOL (*FRConProviderIsWriteContentProgressive)(  
    FS_LPVVOID clientData,  
    FR_Document doc,  
    FS_LPCWSTR lpSaveFilePath  
);
```

**Description**

It will be invoked by Foxit Reader to check whether write the content progressively or all at once.

**Parameter**

---

clientData	[In] The user-supplied data.
doc	[In] It identifies the document opened. It can be used as a unique identifier for the document.
lpSaveFilePath	[In] Foxit Reader passes the file path where the document is saved to the plug-in.

---

**Return**

[TRUE](#) indicates that write the content progressively, then [FRConProviderOnWriteContent](#) will be invoke over and over.

**Head file reference**

fr\_appExpT.h: 3057

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.3](#)

**FRConProviderWriteContentProgressiveFinish**

**Syntax**

```
typedef FS_BOOL (*FRConProviderWriteContentProgressiveFinish)(
    FS_LPVVOID clientData,
    FR_Document doc,
    FS_LPCWSTR lpSaveFilePath,
    FS_BOOL bResult
);
```

**Description**

It will be invoked by Foxit Reader to notify that finish writing content progressively.

**Parameter**

clientData	[In] The user-supplied data.
doc	[In] It identifies the document opened. It can be used as a unique identifier for the document.
lpSaveFilePath	[In] Foxit Reader passes the file path where the document is saved to the plug-in.
bResult	[In] It indicates whether the process of writing content is successful.

**Return**

[TRUE](#) means successful. Otherwise not.

**Head file reference**

fr\_appExpT.h: 3071

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.3](#)

**FRConProviderIsWriteAttachmentContent****Syntax**

```
typedef FS_BOOL (*FRConProviderIsWriteAttachmentContent)(
    FS_LPVVOID clientData,
    FR_Document frDoc,
    FS_LPCWSTR lpAttachmntPath
);
```

**Description**

When the attachment is opened, it will be saved to a temp directory first. This interface will be invoked to determine that whether the plug-in is to write the attachment content itself or not. If this interface returns TRUE, [FRConProviderOnWriteAttachmentContent](#) will be invoked several times to write the attachment content in blocks.

### Parameter

---

clientData	[In] The user-supplied data.
frDoc	[In] It is the document whose attachment is opened.
lpAttachmntPath	[In] The path of the attachment.

---

### Return

[TRUE](#) means the plug-in is to write the attachment content itself, otherwise not.

### Head file reference

fr\_appExpT.h: 3087

### Group

[FR\\_ContentProviderCallbacksRec](#)

### Since

[SDK\\_LATEEST\\_VERSION > 3.0](#)

## FRConProviderOnWriteAttachmentContentFinish

### Syntax

```
typedef FS_BOOL (*FRConProviderOnWriteAttachmentContentFinish)(  
    FS_LVOID clientData,  
    FR_Document frDoc,  
    FS_LPCWSTR lpAttachmntPath  
)
```

### Description

When the attachment is opened, it will be saved to a temp directory first. This interface will be invoked when finishing writing the attachment content to the temp directory.

### Parameter

---

clientData	[In] The user-supplied data.
frDoc	[In] It is the document whose attachment is opened.
lpAttachmntPath	[In] The path of the attachment.

---

**Return**

[TRUE](#) means successful, otherwise failure.

**Head file reference**

fr\_appExpT.h: 3101

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 3.0](#)

**FRCOnProviderOnReadAttachmentContentInBlocks****Syntax**

```
typedef FS_BOOL (*FRCOnProviderOnReadAttachmentContentInBlocks)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc,  
    FS_DWORD pos,  
    unsigned char* pBuf,  
    unsigned long size,  
    FS_LPCWSTR lpAttachmntPath  
);
```

**Description**

When the attachment is modified, the Foxit Reader needs to reattach the attachment. This interface is invoked to get the content of the attachment modified. Firstly, [FRCOnProviderOnGetAttachmentSize](#) will be invoked to obtain the size of the attachment content. Since [SDK\\_LATEEST\\_VERSION > 3.0](#), [FRCOnProviderOnReadAttachmentContent](#) is replaced with [FRCOnProviderOnReadAttachmentContentInBlocks](#).

**Parameter**


---

clientData	[In] The user-supplied data.
frDoc	[In] It is the document whose attachment is opened.
pos	[In] It identifies the position of the document data.
pBuf	[Out] It receives the content of the attachment modified.
size	[In] The total size of the attachment.
lpAttachmntPath	[In] The path of the attachment.

---

**Return**

TRUE means successful, otherwise failure.

**Head file reference**

fr\_appExpT.h: 3120

**Group**

[FR\\_ContentProviderCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 3.0](#)

**FRCOnProviderOnBackFillContent****Syntax**

```
typedef FS_BOOL (*FRCOnProviderOnBackFillContent)(  
    FS_LPVVOID clientData,  
    FR_Document doc,  
    unsigned char* pBuf,  
    unsigned long size,  
    FS_LPCWSTR lpSaveFilePath  
) ;
```

**Description**

It will be invoked by Foxit Reader when some data need to be backfilled after [FRCOnProviderOnWriteAttachmentContentFinish](#) finishes.

**Parameter**

clientData	[In] The user-supplied data.
doc	[In] It identifies the document opened. It can be used as a unique identifier for the document.
pBuf	[In] Foxit Reader passes the document data that is modified to the plug-in.
size	[In] The buffer size.
lpSaveFilePath	[In] Foxit Reader passes the file path where the document is saved to the plug-in.

**Return**

TRUE means successful. Otherwise not.

**Head file reference**

fr\_appExpT.h: 3134

**Group**[FR\\_ContentProviderCallbacksRec](#)**FREnryptCreateHandler****Syntax**

```
typedef FS_LPVOID (*FREnryptCreateHandler)(
    FS\_LPVOID clientData,
    FPD Object encryptDict,
    FS\_LPVOID securityData
);
```

**Description**

Initializes the crypto handler with the data of security handler, and the encrypt dictionary data.

**Parameter**

clientData	[In] The user-supplied data.
encryptDict	[In] The encrypt dictionary
securityData	[In] The data returned from <a href="#">FRSecurityOnInit ()</a> ;

**Return**

Non-zero means initializing successfully, otherwise failed. The data returned can be passed through *cryptoHandler* .

**Head file reference**

fr\_appExpT.h: 2507

**Group**[FR\\_CryptoCallbacksRec](#)**FREnryptDecryptGetSize****Syntax**

```
typedef FS_DWORD (*FREnryptDecryptGetSize)(
    FS\_LPVOID clientData,
    FS\_LPVOID cryptoHandler,
    FS\_DWORD src_size
);
```

**Description**

Estimate size of decrypted data, from a source size.If implementation doesn't want to estimate, it can always return 0.

**Parameter**


---

clientData	[In] The user-supplied data.
cryptoHandler	[In] The data returned from <a href="#">FREncryptCreateHandler()</a> ;
src_size	[In] The input source size.

---

**Return**

The estimated size. 0 means the implementation doesn't want to estimate. If implementation doesn't want to estimate, it can always return 0.

**Head file reference**

fr\_appExpT.h: 2521

**Group**

[FR\\_CryptoCallbacksRec](#)

**FREncryptDecryptStart****Syntax**

```
typedef FS_LPVOID (*FREncryptDecryptStart)(
    FS_LPVOID clientData,
    FS_LPVOID cryptoHandler,
    FS_DWORD objnum,
    FS_DWORD gennum
);
```

**Description**

Starts a decryption process. If decryption is for some indirect object, object number and generationnumber are provided. Implementation can create a context and return pointer to the context.

**Parameter**


---

clientData	[In] The user-supplied data.
cryptoHandler	[In] The data returned from <a href="#">FREncryptCreateHandler()</a> ;
objnum	[In] The indirect object number.
gennum	[In] The indirect object generation number.

---

**Return**

A decrypt context.

#### **Head file reference**

fr\_appExpT.h: 2537

#### **Group**

[FR\\_CryptoCallbacksRec](#)

### FREnryptDecryptStream

#### **Syntax**

```
typedef FS_BOOL (*FREnryptDecryptStream)(  
    FS_LPVVOID clientData,  
    FS_LPVVOID cryptoContext,  
    FS_LPCBYTE src_buf,  
    FS_DWORD src_size,  
    FS_BinaryBuf dest_buf  
)
```

#### **Description**

Decrypt some source data in a stream. The *cryptoContext* parameter is the same as returned by [FREnryptDecryptStart\(\)](#) function. Implementation should append the decrypted data (if any) to the *dest\_buf* dynamic array.

#### **Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

cryptoContext	[In] The decrypt context.
---------------	---------------------------

src_buf	[In] The encrypted source data.
---------	---------------------------------

src_size	[In] The size in bytes of the source data.
----------	--

dest_buf	[Out] It appends the decrypted data to the end.
----------	---

#### **Return**

Non-zero means success, otherwise failure.

#### **Head file reference**

fr\_appExpT.h: 2554

#### **Group**

[FR\\_CryptoCallbacksRec](#)

**FREnryptDecryptFinish****Syntax**

```
typedef FS_BOOL (*FREnryptDecryptFinish)(
    FS_LPVVOID clientData,
    FS_LPVVOID cryptoContext,
    FS_BinaryBuf dest_buf
);
```

**Description**

Finish a decryption process. If decryption context is used, implementation should destroy it. If there is any left-over data, they should be added to the destination buffer.

**Parameter**

clientData	[In] The user-supplied data.
cryptoContext	[In] The decrypt context.
dest_buf	[In] It appends the decrypted data to the end.

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fr\_appExpT.h: 2569

**Group**

[FR\\_CryptoCallbacksRec](#)

**FREnryptEncryptGetSize****Syntax**

```
typedef FS_DWORD (*FREnryptEncryptGetSize)(
    FS_LPVVOID clientData,
    FS_LPVVOID cryptoHandler,
    FS_DWORD objnum,
    FS_DWORD version,
    FS_LPCBYTE src_buf,
    FS_DWORD src_size
);
```

**Description**

Get encrypted data size for a source data block. The returned size should equal to or be larger than the final encrypted data block.

**Parameter**

---

clientData	[In] The user-supplied data.
cryptoHandler	[In] The data returned from <a href="#">FREncryptCreateHandler()</a> ;
objnum	[In] The indirect object number.
version	[In] The indirect object generation number.
src_buf	[In] The source data block.
src_size	[In] The size in bytes of the source data.

---

**Return**

The encrypted data size. It should equal to or be larger than the final encrypted data block.

**Head file reference**

fr\_appExpT.h: 2585

**Group**

[FR\\_CryptoCallbacksRec](#)

**FREncryptEncryptContent****Syntax**

```
typedef FS_BOOL (*FREncryptEncryptContent)(
    FS\_LPVVOID clientData,
    FS\_LPVVOID cryptoHandler,
    FS\_INT32 objnum,
    unsigned long version,
    FS\_LPCBYTE src_buf,
    FS\_DWORD src_size,
    FS\_LPBYTE dest_buf,
    FS\_DWORD* outDestsize
);
```

**Description**

Do the encryption process.Final encrypted data block should be output in the *outDestsize* parameter.

**Parameter**


---

clientData	[In] The user-supplied data.
cryptoHandler	[In] The data returned from <a href="#">FREncryptCreateHandler()</a> ;

---

---

objnum	[In] The indirect object number.
version	[In] The indirect object generation number.
src_buf	[In] The source data block.
src_size	[In] The size in bytes of the source data.
dest_buf	[Out] It receives the encrypted data.
outDestsize	[Out] It receives the size in bytes of the encrypted data.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fr\_appExpT.h: 2603

**Group**

[FR\\_CryptoCallbacksRec](#)

**FREncryptFinishHandler****Syntax**

```
typedef void (*FREncryptFinishHandler)(
    FS\_LPVVOID clientData,
    FS\_LPVVOID cryptoHandler
);
```

**Description**

Release the crypto handler.

**Parameter**


---

clientData	[In] The user-supplied data.
cryptoHandler	[In] The data returned from <a href="#">FREncryptCreateHandler</a> ();

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 2615

**Group**[FR\\_CryptoCallbacksRec](#)**FREncryptProgressiveEncryptStart****Syntax**

```
typedef FS_BOOL (*FREncryptProgressiveEncryptStart)(
    FS_LVOID clientData,
    FS_LVOID cryptoHandler,
    FS_DWORD objnum,
    unsigned long version,
    FS_DWORD raw_size,
    FS_BOOL bFlateEncode
);
```

**Description**

It is invoked by Foxit Reader when encrypting if you set the progressive encrypt handler by [FPDCreatorSetProgressiveEncryptHandler](#) .

**Parameter**

clientData	[In] The user-supplied data.
cryptoHandler	[In] The data returned from <a href="#">FREncryptCreateHandler</a> ();
objnum	[In] The object number.
version	[In] The indirect object generation number.
raw_size	[In] The raw size of stream.
bFlateEncode	[In] Indicates whether the source data need to be encoded.

**Return**

TRUE if you want to encrypt the object stream progressively, otherwise not.

**Head file reference**

fr\_appExpT.h: 2631

**Group**[FR\\_CryptoCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)

**FREnryptoProgressiveEncryptContent****Syntax**

```
typedef FS_BOOL (*FREnryptoProgressiveEncryptContent)(  
    FS_LPVVOID clientData,  
    FS_LPVVOID cryptoHandler,  
    FS_INT32 objnum,  
    unsigned long version,  
    FS_LPCBYTE src_buf,  
    FS_DWORD src_size,  
    FS_BinaryBuf dest_buf  
)
```

**Description**

It is invoked by Foxit Reader over and over when encrypting if you set the progressive encrypt handler by [FPDCreatorSetProgressiveEncryptHandler](#). Implementation should append the encrypted data (if any) to the *dest\_buf* dynamic array.

**Parameter**


---

clientData	[In] The user-supplied data.
cryptoHandler	[In] The data returned from <a href="#">FREnryptoCreateHandler</a> () ;
objnum	[In] The object number.
version	[In] The indirect object generation number.
src_buf	[In] The source data to be encrypted.
src_size	[In] The size of the source data in bytes.
dest_buf	[Out] It appends the encrypted data to the end.

---

**Return**

TRUE if you encrypt the content successfully, otherwise not.

**Head file reference**

fr\_appExpT.h: 2649

**Group**

[FR\\_CryptoCallbacksRec](#)

**Note:** FREnryptoEncryptContent may also be invoked for some objects.

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)**FREnryptoProgressiveEncryptFinish****Syntax**

```
typedef FS_BOOL (*FREnryptoProgressiveEncryptFinish)(
    FS_LPVVOID clientData,
    FS_LPVVOID cryptoHandler,
    FS_BinaryBuf dest_buf
);
```

**Description**

It is invoked by Foxit Reader when finish encrypting content progressively if you set the progressive encrypt handler by [FPDCreatorSetProgressiveEncryptHandler](#). Implementation should append the encrypted data (if any) to the *dest\_buf* dynamic array.

**Parameter**

clientData	[In] The user-supplied data.
cryptoHandler	[In] The data returned from <a href="#">FREnryptoCreateHandler</a> ();
dest_buf	[Out] It appends the encrypted data to the end.

**Return**

TRUE if you encrypt the content successfully, otherwise not.

**Head file reference**

fr\_appExpT.h: 2663

**Group**

[FR\\_CryptoCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)

**FROnCollectNormalData****Syntax**

```
typedef void (*FROnCollectNormalData)(
    FS_LPVVOID clientData,
    FS_LPCWSTR lpwsFunction,
    FS_LPCWSTR lpwsAction,
    FS_LPCWSTR lpwsContent
);
```

**Description**

It is called by Foxit Reader when some action occurs. It is called by Foxit Reader when some action occurs. Then you can implement the callbacks to collect the normal data differs from the bitmap data.

### Parameter

---

clientData	[In] The user-supplied data.
lpwsFunction	[In] The function name whose action occurs. See <a href="#">FRDataCollectionFunctionNames</a> .
lpwsAction	[In] The action name. See <a href="#">FRDataCollectionActionNames</a> .
lpwsContent	[In] The normal data you can collect.

---

### Return

void.

### Head file reference

fr\_appExpT.h: 5462

### Group

[FR\\_DataCollectionHandlerCallbacksRec](#)

### Since

[SDK LATEEST VERSION > 2.1.0.4](#)

## FROnCollectBitmapData

### Syntax

```
typedef void (*FROnCollectBitmapData)(  
    FS_LPVVOID clientData,  
    FS_LPCWSTR lpwsFunction,  
    FS_LPCWSTR lpwsAction,  
    FS_DIBitmap bitmap  
)
```

### Description

It is called by Foxit Reader when some action occurs. It is called by Foxit Reader when some action occurs. Then you can implement the callbacks to collect the bitmap data differs from the normal data.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

lpwsFunction	[In] The function name whose action occurs. See <a href="#">FRDataCollectionFunctionNames</a> .
lpwsAction	[In] The action name. See <a href="#">FRDataCollectionActionNames</a> .
bitmap	[In] The bitmap data you can collect.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5478

**Group**[FR\\_DataCollectionHandlerCallbacksRec](#)**Since**[SDK LATEEST VERSION > 2.1.0.4](#)**FROnCollectNormalData2****Syntax**

```
typedef void (*FROnCollectNormalData2)(  
    FS_LPVOID clientData,  
    FS_LPCWSTR lpwsFunction,  
    FS_LPCWSTR lpwsAction,  
    FS_LPCWSTR lpwsContent,  
    FS_INT32 nLevel  
)
```

**Description**

It is called by Foxit Reader when some action occurs. It is called by Foxit Reader when some action occurs. Then you can implement the callbacks to collect the normal data differs from the bitmap data.

**Parameter**


---

clientData	[In] The user-supplied data.
lpwsFunction	[In] The function name whose action occurs. See <a href="#">FRDataCollectionFunctionNames</a> .
lpwsAction	[In] The action name. See <a href="#">FRDataCollectionActionNames</a> .
lpwsContent	[In] The normal data you can collect.

---

---

nLevel	[In] The detail level of the data.
--------	------------------------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5495

**Group**[FR\\_DataCollectionHandlerCallbacksRec](#)**Since**[SDK LATEEST VERSION > 8.2.1](#)**FROnCollectBitmapData2****Syntax**

```
typedef void (*FROnCollectBitmapData2)(
    FS_LPVVOID clientData,
    FS_LPCWSTR lpwsFunction,
    FS_LPCWSTR lpwsAction,
    FS_DIBitmap bitmap,
    FS_INT32 nLevel
);
```

**Description**

It is called by Foxit Reader when some action occurs. It is called by Foxit Reader when some action occurs. Then you can implement the callbacks to collect the bitmap data differs from the normal data.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

lpwsFunction	[In] The function name whose action occurs. See <a href="#">FRDataCollectionFunctionNames</a> .
--------------	---

---

lpwsAction	[In] The action name. See <a href="#">FRDataCollectionActionNames</a> .
------------	---

---

bitmap	[In] The bitmap data you can collect.
--------	---------------------------------------

---

nLevel	[In] The detail level of the data.
--------	------------------------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5512

**Group**[FR\\_DataCollectionHandlerCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 8.2.1](#)**FRDocPropertyPageOnCreate****Syntax**

```
typedef void (*FRDocPropertyPageOnCreate)(  
    FS\_LPVOID clientData,  
    HWND window  
)
```

**Description**

A callback for document-property-page handler. It is called after the property dialog is created. Client should create a new property page and register it using [FRApAddDocPropertyPage\(\)](#) in this method.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

window	[In] The parent dialog.
--------	-------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 3595

**Group**[FR\\_DocPropertyPageCallbacksRec](#)**FRDocPropertyPageOnDestroy****Syntax**

```
typedef void (*FRDocPropertyPageOnDestroy)(  
    FS\_LPVOID clientData  
)
```

**Description**

A callback for document-property-page handler. It is called when user closes the property dialog. Client should destroy the property page window and free dynamic memory in this method.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 3606

**Group**[FR\\_DocPropertyPageCallbacksRec](#)**FRDocPropertyPageOnSaveData****Syntax**

```
typedef void (*FRDocPropertyPageOnSaveData)(  
    FS\_LPVOID clientData  
)
```

**Description**

A callback for document-property-page handler. It is called when user click the *OK* button on the property dialog.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 3616

**Group**[FR\\_DocPropertyPageCallbacksRec](#)**FRCryptoSetKey****Syntax**

```
typedef void (*FRCryptoSetKey)(  
    FS\_LPVOID clientData,  
    FS\_KeyType ptype,  
    FS\_LPBYTE bsKey,  
    FS\_INT32 pnKeyLen  
)
```

**Description**

Set the key when encryption and decryption.

**Parameter**


---

clientData	[In] The user-supplied data.
ptype	[Out] The type of key to encrypt/decrypt data.
bsKey	[Out] A 16-byte random number.
pnKeyLen	[Out] The length of the <a href="#">bsKey</a> , it always be 16.

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 2764

**Group**[FR\\_DRMCryptoCallbacksRec](#)

**Note:** In the same encryption process, the baKey must be same one.

**Since**[SDK\\_LATEEST\\_VERSION > 9.4](#)**FRCryptoCurrentEncryptObjNum****Syntax**

```
typedef void (*FRCryptoCurrentEncryptObjNum)(
    FS\_LPVVOID clientData,
    FS\_DWORD dwObjNum
);
```

**Description**

Get current encryption PDF object number.

**Parameter**


---

clientData	[In] The user-supplied data.
dwObjNum	[In] Current encrypting object number.

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 2776

**Group**[FR\\_DRMCryptoCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 9.4](#)

FRSecurityOnInit

**Syntax**

```
typedef FS_LPVOID (*FRSecurityOnInit)(  
    FS_LPVOID clientData,  
    FS_LPCWSTR filePath,  
    FPD_Object encryptDict,  
    FPD_Document doc  
)
```

**Description**

It is invoked to initialize the security handler. Plug-ins can initialize the data in this callback. The handler should typically keep the encryption dictionary, and process data in the encryption dictionary.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

filePath	[In] The file path of the PDF document.
----------	---

---

encryptDict	[In] The Encrypt dictionary.
-------------	------------------------------

---

doc	[In] The PDF document.
-----	------------------------

---

**Return**

Non-zero means initializing successfully, otherwise failed. The data returned can be passed through *securityHandler*.

**Head file reference**

fr\_appExpT.h: 2699

**Group**[FR\\_DRMSecurityCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 9.5](#)

## FRSecurityGetPermissions

### Syntax

```
typedef FS_LPVOID (*FRSecurityGetPermissions)(  
    FS_LPVOID clientData,  
    FS_LPVOID securityData,  
    clientData,  
    securityData  
>);
```

### Description

Create a crypto handler that can do the real encryption/decryption work.

[SDK\\_LATEEST\\_VERSION > 9.5](#)

### Parameter

clientData	[In] The user-supplied data.
------------	------------------------------

securityData	[In] The data returned from <a href="#">FRSecurityOnInit()</a> ;
--------------	--

clientData	[In] The user-supplied data.
------------	------------------------------

securityData	[In] The data returned from <a href="#">FRSecurityOnInit()</a> ;
--------------	--

### Return

If *outType* is 0, this callback returns the pointer to structure containing the crypto handler callbacks [FR\\_DRMCryptoCallbacks](#). If *outType* is 1, this callback returns the pointer to internal crypto handler.

### Head file reference

fr\_appExpT.h: 2723

### Group

[FR\\_DRMSecurityCallbacksRec](#)

### Since

[SDK\\_LATEEST\\_VERSION > 9.5](#)

## FRSecurityIsProcessErrMsg

### Syntax

```
typedef FS_BOOL (*FRSecurityIsProcessErrMsg)(  
    FS_LPVOID clientData  
>);
```

**Description**

Checks whether the plug-in processes the error message.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

whether the plug-in processes the error message.

**Head file reference**

fr\_appExpT.h: 2733

**Group**

[FR\\_DRMSecurityCallbacksRec](#)

**FRApEmptyFramWndCanClose****Syntax**

```
typedef FS_BOOL (*FRApEmptyFramWndCanClose)(
    FS_LPVVOID clientData,
    HWND wnd
);
```

**Description**

It is called when user click the "close" button to close a frame which created by [FRApCreateAnEmptyFrameWnd \(\)](#).

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

wnd	[In] The frame window-self which created by <a href="#">FRApCreateAnEmptyFrameWnd ()</a> .
-----	--

---

**Return**

[TRUE](#) if the empty frame can be closed, otherwise not.

**Head file reference**

fr\_appExpT.h: 4463

**Group**

[FR\\_EmpyFramWndNotifiesRec](#)

**FRToMast****Syntax**

```
typedef FS_BOOL (*FRToMast)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc  
);
```

**Description**

Tests whether a document can be controlled to print.

**Parameter**

clientData	[In] The user-supplied data.
frDoc	[In] The document to be printed.

**Return**

If the document can be controlled to print by plug-in, return [TRUE](#) , otherwise [FALSE](#) .

**Head file reference**

fr\_appExpT.h: 3164

**Group**

[FR\\_ExtraPrintInfoProviderCallbackRec](#)

**FRCCanBePrinted****Syntax**

```
typedef FS_BOOL (*FRCCanBePrinted)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 copies,  
    FS_BOOL bTotalPages,  
    FS_DWordArray arrPages,  
    FS_BOOL bPostScript,  
    FS_BOOL bLocal,  
    FS_BOOL bNetWork,  
    FS_BOOL bShared,  
    FS_BytString Printer,  
    FS_BytString PrinterModel,  
    FS_BytString PrinterDriver,  
    FS_BytString PrinterData,  
    FS_BytString PrinterProt  
);
```

**Description**

Tests whether a document can be printed.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The document to be printed.
copies	[In] The number of copies.
bTotalPages	[In] A flag indicates whether all pages will be printed.
arrPages	[In] A integer array use to store the index of page which to be printed. If <i>bTotalPages</i> is <a href="#">TRUE</a> , the array is an empty array.
bPostScript	[In] A flag indicates whether the printer is a post-script printer.
bLocal	[In] A flag indicates whether the printer is a local printer.
bNetWork	[In] A flag indicates whether the printer is a net work printer.
bShared	[In] A flag indicates whether the printer is a shared printer.
Printer	[In] The name of printer.
PrinterModel	[In] The printer model.
PrinterDriver	[In] The printer driver name.
PrinterData	[In] The printer data.
PrinterProt	[In] The print port.

---

**Return**

If the document can be printed, return [TRUE](#), otherwise [FALSE](#).

**Head file reference**

fr\_appExpT.h: 3188

**Group**

[FR\\_ExtraPrintInfoProviderCallbackRec](#)

**FRCOUNTOWNERTEXTPRINTINFO****Syntax**

```
typedef FS_INT32 (*FRCOUNTOWNERTEXTPRINTINFO)(
```

---

```
FS_LPVOID clientData,
FR_Document frDoc,
FS_INT32 iPage
);
```

**Description**

Gets the total number of text information which will be printed as a watermark of specified page.

**Parameter**

clientData	[In] The user-supplied data.
frDoc	[In] The document which to be printed.
iPage	[In] The page index indicates the page which will be printed.

**Return**

A integer indicates total number of the text information which will be printed as a watermark in a page.

**Head file reference**

fr\_appExpT.h: 3216

**Group**

[FR\\_ExtraPrintInfoProviderCallbackRec](#)

**FRCOUNTOWNERIMGPRINTINFO****Syntax**

```
typedef FS_INT32 (*FRCOUNTOWNERIMGPRINTINFO)(
    FS_LPVOID clientData,
    FR_Document frDoc,
    FS_INT32 iPage
);
```

**Description**

Gets the total number of image information which will be printed as a watermark of specified page.

**Parameter**

clientData	[In] The user-supplied data.
frDoc	[In] The document which to be printed.

---

iPage	[In] The page index indicates the page which will be printed.
-------	---

---

**Return**

A integer indicates total number of the image information which will be printed as a watermark in a page.

**Head file reference**

fr\_appExpT.h: 3229

**Group**

[FR\\_ExtraPrintInfoProviderCallbackRec](#)

**FRPrintOwnerTextInfo****Syntax**

```
typedef FS_ByteString (*FRPrintOwnerTextInfo)(
    FS_LPVVOID clientData,
    FR_Document frDoc,
    FS_INT32 iPage,
    FS_INT32 nStartX,
    FS_INT32 nStartY,
    FS_INT32 nWidth,
    FS_INT32 nHeight,
    FS_INT32 index,
    FS_INT32* textLeftPos,
    FS_INT32* textTopPos,
    FS_FLOAT* textOpacity,
    FS_DWORD* textColor,
    void* textUseFont,
    FS_AffineMatrix* textMatrix
);
```

**Description**

Passes the text information to Foxit Reader to print as a watermark of a page.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---



---

frDoc	[In] The document which to be printed.
-------	--

---



---

iPage	[In] The page index.
-------	----------------------

---



---

nStartX	[In] Left pixel position of the page specified by <i>iPage</i> .
---------	--

---



---

nStartY	[In] Top pixel position of the page specified by <i>iPage</i> .
---------	---

---

---

nWidth	[In] Horizontal size (in pixels) of the page specified by <i>iPage</i> .
nHeight	[In] Vertical size (in pixels) of the page specified by <i>iPage</i> .
index	[In] The zero based index of text information.
textLeftPos	[Out] Left pixel position of the display area in the device coordination.
textTopPos	[Out] Top pixel position of the display area in the device coordination.
textOpacity	[Out] The opacity of the information.
textColor	[Out] The color of the text.
textUseFont	[Out] The font of the text.
textMatrix	[Out] The matrix of the text.

---

**Return**

A byte string which storing the text, and the string will be printed as a watermark with a page.

**Head file reference**

fr\_appExpT.h: 3253

**Group**

[FR\\_ExtraPrintInfoProviderCallbackRec](#)

**FRPrintOwnerImgInfo**

**Syntax**

```
typedef FS_DIBitmap (*FRPrintOwnerImgInfo)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 iPage,  
    FS_INT32 nStartX,  
    FS_INT32 nStartY,  
    FS_INT32 nWidth,  
    FS_INT32 nHeight,  
    FS_INT32 index,  
    FS_INT32* imgLeftPos,  
    FS_INT32* imgTopPos,  
    FS_FLOAT* imgOpacity,  
    FS_AffineMatrix* imgMatrix
```

);

### Description

#### Parameter

clientData	[In] The user-supplied data.
frDoc	[In] The document which to be printed.
iPage	[In] The page index.
nStartX	[In] Left pixel position of the page specified by <i>iPage</i> .
nStartY	[In] Top pixel position of the page specified by <i>iPage</i> .
nWidth	[In] Horizontal size (in pixels) of the page specified by <i>iPage</i> .
nHeight	[In] Vertical size (in pixels) of the page specified by <i>iPage</i> .
index	[In] The zero based index of text information.
imgLeftPos	[Out] Left pixel position of the display area in the device coordination.
imgTopPos	[Out] Top pixel position of the display area in the device coordination.
imgOpacity	[Out] The opacity used to show image.
imgMatrix	[Out] The matrix used to show image.

#### Return

A bitmap which will be printed as a watermark with a page.

#### Head file reference

fr\_appExpT.h: 3289

#### Group

[FR\\_ExtraPrintInfoProviderCallbackRec](#)

**FRPrintOnPreviewDidRender****Syntax**

```
typedef void (*FRPrintOnPreviewDidRender)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc,  
    FPD_Page pdfPage,  
    FS_INT32 iPage,  
    FS_AffineMatrix matrix,  
    FPD_RenderDevice renderDevice  
)
```

**Description**

It is invoked by Foxit Reader when the page has been rendered to preview dialog. You can use the *renderDevice* to render extra content to the preview dialog.

**Parameter**


---

clientData	[In] The user-supplied data.
frDoc	[In] The document which to be printed.
pdfPage	[In] The Page to be rendered.
iPage	[In] The page index.
matrix	[In] The matrix used to render the page to device.
renderDevice	[In] The memory device used to render the page.

---

**Return****Head file reference**

fr\_appExpT.h: 3317

**Group**

[FR\\_ExtraPrintInfoProviderCallbackRec](#)

**FRPrintOnDidRender****Syntax**

```
typedef void (*FRPrintOnDidRender)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc,  
    FPD_Page pdfPage,  
    FS_INT32 iPage,  
    FS_AffineMatrix matrix,
```

```
FPD\_RenderDevice renderDevice  
);
```

**Description**

It is invoked by Foxit Reader when the page has been printed to the printer. You can use the *renderDevice* to print extra content to the printer.

**Parameter**

clientData	[In] The user-supplied data.
frDoc	[In] The document which to be printed.
pdfPage	[In] The Page to be rendered.
iPage	[In] The page index.
matrix	[In] The matrix used to render the page to device.
renderDevice	[In] The memory device used to render the page.

**Return****Head file reference**

fr\_appExpT.h: 3338

**Group**

[FR\\_ExtraPrintInfoProviderCallbackRec](#)

**FRPrintIsPrintOpacity****Syntax**

```
typedef FS_BOOL (*FRPrintIsPrintOpacity)(  
    FS\_LPVVOID clientData,  
    FR\_Document frDoc,  
    FPD\_Page pdfPage,  
    FS\_INT32 iPage  
)
```

**Description**

It is invoked by Foxit Reader before the page is rendered to memory device.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The document which to be printed.
-------	--

---

pdfPage	[In] The Page to be rendered.
---------	-------------------------------

---

iPage	[In] The page index.
-------	----------------------

---

**Return**

Whether to print opacity content.

**Head file reference**

fr\_appExpT.h: 3357

**Group**

[FR\\_ExtraPrintInfoProviderCallbackRec](#)

**FRCCanBePrinted2****Syntax**

```
typedef FS_BOOL (*FRCCanBePrinted2)(  
    FS_LVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 copies,  
    FS_BOOL bTotalPages,  
    FS_DWordArray arrPages,  
    FS_BOOL bPostScript,  
    FS_BOOL bLocal,  
    FS_BOOL bNetWork,  
    FS_BOOL bShared,  
    FS_WideString Printer,  
    FS_BytString PrinterModel,  
    FS_BytString bsSubset,  
    FS_BytString PrinterDriver,  
    FS_BytString PrinterData,  
    FS_BytString PrinterProt  
) ;
```

**Description**

Tests whether a document can be printed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The document to be printed.
-------	----------------------------------

---

copies	[In] The number of copies.
--------	----------------------------

---

---

bTotalPages	[In] A flag indicates whether all pages will be printed.
arrPages	[In] A integer array use to store the index of page which to be printed. If <i>bTotalPages</i> is <a href="#">TRUE</a> ,the array is an empty array.
bPostScript	[In] A flag indicates whether the printer is a post-script printer.
bLocal	[In] A flag indicates whether the printer is a local printer.
bNetWork	[In] A flag indicates whether the printer is a net work printer.
bShared	[In] A flag indicates whether the printer is a shared printer.
Printer	[In] The name of printer.
PrinterModel	[In] The printer model.
bsSubset	[In] The subset.
PrinterDriver	[In] The printer driver name.
PrinterData	[In] The printer data.
PrinterProt	[In] The print port.

---

**Return**

If the document can be printed, return [TRUE](#) , otherwise [FALSE](#) .

**Head file reference**

fr\_appExpT.h: 3386

**Group**

[FR\\_ExtraPrintInfoProviderCallbackRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRCaptureGetType****Syntax**

```
typedef FS_LPSTR (*FRCaptureGetType)(
```

```
FS_LPVVOID clientData  
);
```

**Description**

It is called to get the mouse point handler type.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

The type of mouse point handler.

**Head file reference**

fr\_appExpT.h: 4226

**Group**

[FR\\_MousePtCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRMousePtGetObjectAtPoint****Syntax**

```
typedef void* (*FRMousePtGetObjectAtPoint)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    FS_DevicePoint point  
);
```

**Description**

It is called to get the object at the specified point.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

pageView	[In] The input page view whose left-button-down event occurred.
----------	---

---

point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.
-------	---

---

**Return**

The object at the specified point.

**Head file reference**

fr\_appExpT.h: 4239

**Group**[FR\\_MousePtCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRMousePtLButtonDown****Syntax**

```
typedef FS_BOOL (*FRMousePtLButtonDown)(  
    FS\_LVOID clientData,  
    FR\_PageView pageView,  
    void* curData,  
    unsigned int nFlags,  
    FS\_DevicePoint point  
)
```

**Description**

It is called by Foxit Reader to notify the left-button-down event.

**Parameter**

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose left-button-down event occurred.
curData	[In] The current data for doc.
nFlags	[In] Indicates whether various virtual keys are down.
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

**Return**

[TRUE](#) if the left-button-down event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 4254

**Group**[FR\\_MousePtCallbacksRec](#)

**Since**  
[SDK LATEEST VERSION > 2.0](#)

## FRMousePtLButtonUp

### Syntax

```
typedef FS_BOOL (*FRMousePtLButtonUp)(  
    FS\_LPVVOID clientData,  
    FR\_PageView pageView,  
    void* curData,  
    unsigned int nFlags,  
    FS\_DevicePoint point  
) ;
```

### Description

It is called by Foxit Reader to notify the left-button-up event.

### Parameter

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose left-button-up event occurred.
curData	[In] The current data for doc.
nFlags	[In] Indicates whether various virtual keys are down. Reference to <i>CWnd::OnLButtonUp</i> .
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

### Return

[TRUE](#) if the left-button-up event was handled, otherwise not.

### Head file reference

fr\_appExpT.h: 4269

### Group

[FR\\_MousePtCallbacksRec](#)

### Since

[SDK LATEEST VERSION > 2.0](#)

## FRMousePtLButtonDblClk

**Syntax**

```
typedef FS_BOOL (*FRMousePtLButtonDblClk)(  
    FS\_LPVVOID clientData,  
    FR\_PageView pageView,  
    void* curData,  
    unsigned int nFlags,  
    FS\_DevicePoint point  
)
```

**Description**

It is called by Foxit Reader to notify the left-button-double-click event.

**Parameter**


---

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose left-button-double-click event occurred.
curData	[In] The current data for doc.
nFlags	[In] Indicates whether various virtual keys are down. Reference to <i>CWnd::OnLButtonDblClk</i> .
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

---

**Return**

[TRUE](#) if the left-button-double-click event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 4284

**Group**

[FR\\_MousePtCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRMousePtMouseMove****Syntax**

```
typedef FS_BOOL (*FRMousePtMouseMove)(  
    FS\_LPVVOID clientData,  
    FR\_PageView pageView,  
    void* curData,  
    unsigned int nFlags,
```

---

```
FS_DevicePoint point
);
```

**Description**

It is called by Foxit Reader to notify the mouse-move event.

**Parameter**

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose mouse-move event occurred.
curData	[In] The current data for doc.
nFlags	[In] Indicates whether various virtual keys are down. Reference to <i>CWnd::OnMouseMove</i> .
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

**Return**

[TRUE](#) if the mouse-move event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 4299

**Group**

[FR\\_MousePtCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRMousePtRButtonDown****Syntax**

```
typedef FS_BOOL (*FRMousePtRButtonDown)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    void* curData,  
    unsigned int nFlags,  
    FS_DevicePoint point  
)
```

**Description**

It is called by Foxit Reader to notify the right-button-down event.

**Parameter**


---

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose right-button-down event occurred.
curData	[In] The current data for doc.
nFlags	[In] Indicates whether various virtual keys are down. Reference to <i>CWnd::OnRButtonDown</i> .
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

---

**Return**

[TRUE](#) if the right-button-down event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 4314

**Group**

[FR\\_MousePtCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRMousePtRButtonUp****Syntax**

```
typedef FS_BOOL (*FRMousePtRButtonUp)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    void* curData,  
    unsigned int nFlags,  
    FS_DevicePoint point  
)
```

**Description**

It is called by Foxit Reader to notify the right-button-up event.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

pageView	[In] The input page view whose right-button-up event occurred.
----------	--

---

curData	[In] The current data for doc.
---------	--------------------------------

---

nFlags	[In] Indicates whether various virtual keys are down. Reference to <i>CWnd::OnRButtonUp</i> .
--------	---

---

point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.
-------	---

---

**Return**

[TRUE](#) if the right-button-up event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 4329

**Group**

[FR\\_MousePtCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRMousePtRButtonDblClk****Syntax**

```
typedef FS_BOOL (*FRMousePtRButtonDblClk)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    void* curData,  
    unsigned int nFlags,  
    FS_DevicePoint point  
>;
```

**Description**

It is called by Foxit Reader to notify the right-button-double-click event.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

pageView	[In] The input page view whose right-button-double-click event occurred.
----------	--

---

curData	[In] The current data for doc.
---------	--------------------------------

---

---

nFlags	[In] Indicates whether various virtual keys are down. Reference to <i>CWnd::OnRButtonDblClk</i> .
--------	---

---

point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.
-------	---

---

**Return**

[TRUE](#) if the right-button-double-click event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 4344

**Group**

[FR\\_MousePtCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRMousePtMouseWheel****Syntax**

```
typedef FS_BOOL (*FRMousePtMouseWheel)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    void* curData,  
    unsigned int nFlags,  
    short zDelta,  
    FS_DevicePoint point  
) ;
```

**Description**

It is called by Foxit Reader to notify the mouse-wheel event.

**Parameter**


---

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose mouse-wheel event occurred.
curData	[In] The current data for doc.
nFlags	[In] Indicates whether various virtual keys are down. Reference to <i>CWnd::OnMouseWheel</i> .

---

zDelta	[In] Indicates distance rotated. Reference to <i>CWnd::OnMouseWheel</i> .
--------	---

---

---

point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.
-------	---

---

**Return**

[TRUE](#) if the mouse-wheel event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 4360

**Group**

[FR\\_MousePtCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRMousePtOnMouseEnter****Syntax**

```
typedef void (*FRMousePtOnMouseEnter)(  
    FS\_LPVVOID clientData,  
    FR\_PageView pageView,  
    void* curData  
)
```

**Description**

It is called by Foxit Reader to notify the mouse-enter event.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---



---

pageView	[In] The input page view whose mouse-enter event occurred.
----------	--

---



---

curData	[In] The current data for doc.
---------	--------------------------------

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 4373

**Group**

[FR\\_MousePtCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRMousePtOnMouseEvent

### Syntax

```
typedef void (*FRMousePtOnMouseEvent)(  
    FS\_LPVOID clientData,  
    FR\_PageView pageView,  
    void* curData  
);
```

### Description

It is called by Foxit Reader to notify the mouse-exit event.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

pageView	[In] The input page view whose mouse-exit event occurred.
----------	---

---

curData	[In] The current data for doc.
---------	--------------------------------

---

### Return

void

### Head file reference

[fr\\_appExpT.h: 4386](#)

### Group

[FR\\_MousePtCallbacksRec](#)

### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FROwnerFileTypeHandlerDoOpen

### Syntax

```
typedef FS_BOOL (*FROwnerFileTypeHandlerDoOpen)(  
    FS\_LPVOID clientData,  
    FS\_LPCSTR lpszFilterName,  
    FS\_LPCWSTR lpszPath  
);
```

### Description

It will be invoked by Foxit Reader when a document is opened. You can implement the callback to control the process of opening.

### Parameter

---

clientData	[In] The user-supplied data.
lpszFilterName	[In] The filter name used in file open dialog.
lpszPath	[In] The full path of the document opened.

---

**Return**

[TRUE](#) if the plug-in takes over the process, otherwise not.

**Head file reference**

fr\_appExpT.h: 4701

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FROwnerFileTypeHandlerDoSave****Syntax**

```
typedef FS_BOOL (*FROwnerFileTypeHandlerDoSave)(  
    FS_LPVVOID clientData,  
    FS_LPCSTR lpszFilterName,  
    FS_LPCWSTR lpszPath  
>;
```

**Description**

It will be invoked by Foxit Reader when a document is saved. You can implement the callback to control the process of saving.

**Parameter**


---

clientData	[In] The user-supplied data.
lpszFilterName	[In] The filter name used in file save dialog.
lpszPath	[In] The full path of the document to be saved.

---

**Return**

[TRUE](#) if the plug-in takes over the process.

**Head file reference**

fr\_appExpT.h: 4715

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FROwnerFileTypeHandlerDoEmail****Syntax**

```
typedef FS_BOOL (*FROwnerFileTypeHandlerDoEmail)(
    FS_LPVVOID clientData,
    FS_LPCSTR lpszFilterName,
    FS_LPCWSTR lpszPath
);
```

**Description**

It will be invoked by Foxit Reader when a document is emailed. You can implement the callback to control the process of sending email.

**Parameter**

clientData	[In] The user-supplied data.
lpszFilterName	[In] The filter name used as the unique identity.
lpszPath	[In] The full path of the document to be emailed.

**Return**

Return [TRUE](#) if the plug-in takes over the process.

**Head file reference**

fr\_appExpT.h: 4729

**Group**[FR\\_OwnerFileTypeHandlerCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FROwnerFileTypeHandlerCanSetFileAssociation****Syntax**

```
typedef FS_BOOL (*FROwnerFileTypeHandlerCanSetFileAssociation)(
    FS_LPVVOID clientData,
    FS_LPCWSTR lpszFileExt
);
```

**Description**

It will be invoked by Foxit Reader to judge whether the plug-in need to associate the file type with Foxit Reader.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

lpszFileExt	[In] The extension of the file.
-------------	---------------------------------

---

**Return**

[TRUE](#) if the file type needs to be associated with Foxit Reader as the default application. If the plug-in returns [FALSE](#) , [FROwnerFileTypeHandlerSetFileAssociationInfo](#) will not be invoked.

**Head file reference**

fr\_appExpT.h: 4742

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FROwnerFileTypeHandlerSetFileAssociationInfo

**Syntax**

```
typedef FS_BOOL (*FROwnerFileTypeHandlerSetFileAssociationInfo)(  
    FS_LPVOID clientData,  
    FS_LPCWSTR lpszFileExt,  
    FS_WideString wsRegClass,  
    FS_WideString wsDesc,  
    FS_WideString wsAppPath,  
    FS_INT32* nDefaultIcon  
) ;
```

**Description**

It will be invoked by Foxit Reader to set the file association information.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

lpszFileExt	[In] The extension of the file.
-------------	---------------------------------

---

---

wsRegClass	[Out] It receives the application object name under <i>HKEY_CLASSES_ROOT</i> . For example, "FoxitPhantomPDF.Document".
wsDesc	[Out] It receives the description of the application object.
wsAppPath	[Out] It receives the application path where the icon is stored.

---

nDefaultIcon	[Out] The default icon index displayed on the desktop for owner file type. You can set it 1 as default.Put the icon resource in the RC file of your plug-in and set the ID value to 102.
--------------	--

---

**Return**

[TRUE](#) if the plug-in set the file association information correctly, otherwise not.

**Head file reference**

fr\_appExpT.h: 4759

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FROwnerFileTypeHandlerGetSupportFileTypeCount****Syntax**

```
typedef FS_INT32 (*FROwnerFileTypeHandlerGetSupportFileTypeCount)(
    FS_LPVOID clientData
);
```

**Description**

It will be invoked by Foxit Reader to get the count of file types that this handler can support.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

The count of file types that this handler can support.

**Head file reference**

fr\_appExpT.h: 4771

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Since**[SDK LATEEST VERSION > 1.0](#)**FROwnerFileTypeHandlerGetFileNameByIndex****Syntax**

```
typedef FS_BOOL (*FROwnerFileTypeHandlerGetFileNameByIndex)(
    FS\_LPVOID clientData,
    FS\_INT32 nIndex,
    FS\_ByteString bsName
);
```

**Description**

It will be invoked by Foxit Reader to get the filter name of custom file type by index.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

nIndex	[In] The index of the custom file type.
--------	---

bsName	[Out] It receives the filter name of custom file type.
--------	--

**Return**

[TRUE](#) if gets the filter name correctly, otherwise not.

**Head file reference**

fr\_appExpT.h: 4784

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Note:** eg: Word 97-2003 Files(\*.doc)|\*.doc (must not translate).

**Since**[SDK LATEEST VERSION > 1.0](#)**FROwnerFileTypeHandlerGetFileTypeFilter****Syntax**

```
typedef FS_BOOL (*FROwnerFileTypeHandlerGetFileTypeFilter)(
    FS\_LPVOID clientData,
    FS\_LPCSTR lpszFilterName,
    FS\_WideString wsName
);
```

**Description**

It will be invoked by Foxit Reader to get the filter of the custom file type.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

lpszFilterName	[In] The filter name.
----------------	-----------------------

---

wsName	[Out] It receives the filter of the custom file type.
--------	---

---

**Return**

[TRUE](#) if gets the filter correctly, otherwise not.

**Head file reference**

fr\_appExpT.h: 4797

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Note:** eg: Word 97-2003 Files(\*.doc)|\*.doc (must translate).

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FROwnerFileTypeHandlerGetFileExt****Syntax**

```
typedef FS_BOOL (*FROwnerFileTypeHandlerGetFileExt)(  
    FS\_LPVVOID clientData,  
    FS\_LPCSTR lpszFilterName,  
    FS\_WideString wsFileExt  
>;
```

**Description**

It will be invoked by Foxit Reader to get the extension of the custom file type.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

lpszFilterName	[In] The filter name.
----------------	-----------------------

---

---

wsFileExt	[Out] It receives the extension of the custom file type.
-----------	--

---

**Return**

TRUE if gets the extension correctly, otherwise not.

**Head file reference**

fr\_appExpT.h: 4810

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FROwnerFileTypeHandlerIsEnableSaveAsSettingBtn****Syntax**

```
typedef FS_BOOL (*FROwnerFileTypeHandlerIsEnableSaveAsSettingBtn)(
    FS_LPVVOID clientData,
    FS_LPCSTR lpszFilterName
);
```

**Description**

It will be invoked by Foxit Reader to determine whether enables the setting button in the Save As dialog.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---



---

lpszFilterName	[In] The filter name.
----------------	-----------------------

---

**Return**

TRUE if you want to enable the setting button, otherwise not.

**Head file reference**

fr\_appExpT.h: 4822

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Note:**For Save As Interface

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FROwnerFileTypeHandlerExecuteSaveAsSetting****Syntax**

```
typedef FS_BOOL (*FROwnerFileTypeHandlerExecuteSaveAsSetting)(
    FS_LPVVOID clientData,
    FS_LPCSTR lpszFilterName,
    HWND hParentWnd
);
```

**Description**

It will be invoked by Foxit Reader when the user clicks the setting button.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

lpszFilterName	[In] The filter name.
----------------	-----------------------

hParentWnd	[In] The parent window.
------------	-------------------------

**Return**

[TRUE](#) if the plug-in takes over the setting process, otherwise not.

**Head file reference**

fr\_appExpT.h: 4835

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Note:** For Save As Interface

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FROwnerFileTypeHandlerCanSupportAddToRecentList****Syntax**

```
typedef FS_BOOL (*FROwnerFileTypeHandlerCanSupportAddToRecentList)(
    FS_LPVVOID clientData
);
```

**Description**

It will be invoked by Foxit Reader to determine whether the file of this type can be added to the recent file list.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

[TRUE](#) if the file of this type can be added to the recent file list, otherwise not.

**Head file reference**

fr\_appExpT.h: 4846

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FROwnerFileTypeHandlerCanSupportSave****Syntax**

```
typedef FS_BOOL (*FROwnerFileTypeHandlerCanSupportSave)(
    FS_LPVVOID clientData,
    FS_WideString wsFilterName,
    FR_Document frDoc
);
```

**Description**

It will be invoked by Foxit Reader to determine whether the filter of the document can support to save.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---



---

wsFilterName	[In] The filter name.
--------------	-----------------------

---



---

frDoc	[In] The document.
-------	--------------------

---

**Return**

[TRUE](#) if support the filter, otherwise not.

**Head file reference**

fr\_appExpT.h: 4859

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FROwnerFileTypeHandlerCanBeSaveAsToOtherExt

### Syntax

```
typedef FS_BOOL (*FROwnerFileTypeHandlerCanBeSaveAsToOtherExt)(  
    FS_LPVVOID clientData,  
    FS_WideString wsFilterName  
)
```

### Description

It will be invoked by Foxit Reader to determine whether the filter can be save as to other extensions.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

wsFilterName	[In] The filter name.
--------------	-----------------------

---

### Return

[TRUE](#) if support to save as other extensions, otherwise not.

### Head file reference

fr\_appExpT.h: 4871

### Group

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

### Since

[SDK LATEEST VERSION > 1.0](#)

## FROwnerFileTypeHandlerCanSupportOpen

### Syntax

```
typedef FS_BOOL (*FROwnerFileTypeHandlerCanSupportOpen)(  
    FS_LPVVOID clientData,  
    FS_WideString wsFilterName,  
    FS_WideString wsPathName  
)
```

### Description

It will be invoked by Foxit Reader to determine whether the filter of the document can support to open.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

wsFilterName	[In] The filter name.
--------------	-----------------------

---

wsPathName	[In] The path of the document to be opened.
------------	---

---

**Return**

[TRUE](#) if support the filter, otherwise not.

**Head file reference**

fr\_appExpT.h: 4884

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FROwnerFileTypeHandlerDoOpenDir****Syntax**

```
typedef FS_BOOL (*FROwnerFileTypeHandlerDoOpenDir)(  
    FS\_LVOID clientData,  
    HWND csHandler  
);
```

**Description**

It will be invoked by Foxit Reader to open a dir which includes the opening epub document.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

csHandler	[In] handler of the window for opening epub document.
-----------	---

---

**Return**

[TRUE](#) if operation succeeded, otherwise not.

**Head file reference**

fr\_appExpT.h: 4896

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 8.2](#)

**FROwnerFileTypeHandlerCanSupportOpenDir****Syntax**

```
typedef FS_BOOL (*FROwnerFileTypeHandlerCanSupportOpenDir)(  
    FS_LPVVOID clientData,  
    HWND csHandler  
>;
```

**Description**

It will be invoked by Foxit Reader to determine whether the filter of the document can support to open dir for this current epub document.

**Parameter**


---

clientData	[In] The user-supplied data.
csHandler	[In] handler of the window for opening epub document.

---

**Return**

[TRUE](#) if support the filter, otherwise not.

**Head file reference**

fr\_appExpT.h: 4908

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 8.2](#)

**FROwnerFileTypeHandlerSaveFilePathToClipboard****Syntax**

```
typedef FS_BOOL (*FROwnerFileTypeHandlerSaveFilePathToClipboard)(  
    FS_LPVVOID clientData,  
    HWND csHandler  
>;
```

**Description**

It will be invoked by Foxit Reader to copy current epub document path to clip board.

**Parameter**


---

clientData	[In] The user-supplied data.
csHandler	[In] handler of the window for opening epub document.

---

**Return**

[TRUE](#) if operation successed, otherwise not.

**Head file reference**

fr\_appExpT.h: 4920

**Group**

[FR\\_OwnerFileTypeHandlerCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 8.2](#)

**FRDocWillOpen****Syntax**

```
typedef void (*FRDocWillOpen)(  
    FS\_LPVOID clientData,  
    FR\_Document doc  
)
```

**Description**

A callback for document-level event handler. It is called whenever a document is opening.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] The <a href="#">FR_Document</a> object that is opening.
-----	--

---

**Return****Head file reference**

fr\_appExpT.h: 1127

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRDocDidOpen****Syntax**

```
typedef void (*FRDocDidOpen)(  
    FS\_LPVOID clientData,  
    FR\_Document doc  
)
```

**Description**

A callback for document-level event handler. It is called whenever a document is opened completely.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] The <a href="#">FR Document</a> that is opened completely.
-----	---

---

**Return****Head file reference**

fr\_appExpT.h: 1140

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocOnActivate****Syntax**

```
typedef void (*FRDocOnActivate)(  
    FS\_LPVOID clientData,  
    FR Document doc  
)
```

**Description**

A callback for document-level event handler. It is called whenever a document is to be active.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] The <a href="#">FR Document</a> that becomes an active document.
-----	---

---

**Return****Head file reference**

fr\_appExpT.h: 1153

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocOnDeactivate****Syntax**

```
typedef void (*FRDocOnDeactivate)(  
    FS\_LPVOID clientData,  
    FR Document doc
```

);

**Description**

A callback for document-level event handler. It is called whenever a document is to be deactivate.

**Parameter**

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> that becomes an deactivate document.

**Return**

void

**Head file reference**

fr\_appExpT.h: 1167

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRDocWillSave****Syntax**

```
typedef void (*FRDocWillSave)(  
    FS_LPVOID clientData,  
    FR_Document doc,  
    FS_BOOL bSaveAs  
)
```

**Description**

A callback for document-level event handler. It is called when user clicked the save button on the save dialog. Client should update some data to document.

**Parameter**

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> that will be saved.
bSaveAs	[In] A flag indicates whether user click the "save" button or click the "save as" button.

**Return****Head file reference**

fr\_appExpT.h: 1181

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocDidSave****Syntax**

```
typedef void (*FRDocDidSave)(  
    FS\_LPVOID clientData,  
    FR\_Document doc,  
    FS\_BOOL bSaveAs  
)
```

**Description**

A callback for document-level event handler. It is called when a document is saved completely.

**Parameter**

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> that is saved completely.
bSaveAs	[In] A flag indicates whether user click the "save" button or click the "save as" button.

**Return****Head file reference**

fr\_appExpT.h: 1195

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocWillClose****Syntax**

```
typedef void (*FRDocWillClose)(  
    FS\_LPVOID clientData,  
    FR\_Document doc  
)
```

**Description**

A callback for document-level event handler. It is called when user click the close menu item or close button. Client should do some corresponding operation here.

**Parameter**

---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> that will be closed.

---

**Return****Head file reference**

fr\_appExpT.h: 1208

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocDidClose****Syntax**

```
typedef void (*FRDocDidClose)(
    FS\_LPVOID clientData,
    FR\_Document doc
);
```

**Description**

A callback for document-level event handler. It is called when the document is closed completely.

**Parameter**


---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> that is closed.

---

**Return****Head file reference**

fr\_appExpT.h: 1221

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocDidCopy****Syntax**

```
typedef void (*FRDocDidCopy)(
    FS\_LPVOID clientData,
    FR\_Document doc
);
```

**Description**

A callback for document-level event handler. It is called after some page content has been copied to clipboard.

**Parameter**

---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR Document</a> which content has been copied to clipboard.

---

**Return****Head file reference**

fr\_appExpT.h: 1234

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRDocWillPrint****Syntax**

```
typedef void (*FRDocWillPrint)(  
    FS\_LPVOID clientData,  
    FR Document doc  
)
```

**Description**

A callback for document-level event handler. It is called when the printer start to work.

**Parameter**

---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR Document</a> to be printed.

---

**Return****Head file reference**

fr\_appExpT.h: 1247

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRDocDidPrint**

**Syntax**

```
typedef void (*FRDocDidPrint)(  
    FS\_LPVOID clientData,  
    FR\_Document doc  
)
```

**Description**

A callback for document-level event handler. It is called whenever the printer finished work.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] The <a href="#">FR_Document</a> that is printed completely.
-----	--

---

**Return****Head file reference**

fr\_appExpT.h: 1260

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRDocOnChange****Syntax**

```
typedef void (*FRDocOnChange)(  
    FS\_LPVOID clientData,  
    FR\_Document doc  
)
```

**Description**

A callback for document-level event handler. It is called when a document is modified.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] The <a href="#">FR_Document</a> that is modified.
-----	--

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 1274

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocOnPermissionChange****Syntax**

```
typedef void (*FRDocOnPermissionChange)(  
    FS\_LPVVOID clientData,  
    FR\_Document doc  
);
```

**Description**

A callback for document-level event handler. It is called after the permission of a document is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> whose permission is changed.

---

**Return****Head file reference**

fr\_appExpT.h: 1287

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocWillDraw****Syntax**

```
typedef void (*FRDocWillDraw)(  
    FS\_LPVVOID clientData,  
    FR\_DocView docView,  
    HDC dc  
);
```

**Description**

A callback for document-level event handler. It is called at the beginning of the drawing.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

docView	[In] The <a href="#">FR_DocView</a> that will be drawn.
---------	---

---

dc	[In] The device context of <i>docView</i> .
----	---

---

## Return

### Head file reference

fr\_appExpT.h: 1301

### Group

[FR\\_PageEventCallbacksRec](#)

## FRDocDidDraw

### Syntax

```
typedef void (*FRDocDidDraw)(  
    FS\_LPVOID clientData,  
    FR\_DocView docView,  
    HDC dc  
)
```

### Description

A callback for document-level event handler. It is called when Reader finished the drawing of a document view.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

docView	[In] The <a href="#">FR_DocView</a> that has been drawn.
---------	--

---

dc	[In] The device context of <i>docView</i> .
----	---

---

## Return

### Head file reference

fr\_appExpT.h: 1315

### Group

[FR\\_PageEventCallbacksRec](#)

## FRDocOnWillInsertPages

### Syntax

```
typedef void (*FRDocOnWillInsertPages)(  
    FS\_LPVOID clientData,
```

---

```
FR_Document doc,
FS_INT32 iInsertAt,
FS_INT32 nPages
);
```

**Description**

A callback for document-level event handler. It is called to notify all client that the number of page views will increase. At this time, the number of FPD\_Page may be increased, meaning there is some FPD\_Page inserted to the FPD\_Document before this calling.

**Parameter**


---

clientData	[In] The user-supplied data.
doc	[In] The <u>FR_Document</u> whose page views will be inserted.
iInsertAt	[In] The insert point where the page will be inserted.
nPages	[In] The page number of inserted page.

---

**Return****Head file reference**

fr\_appExpT.h: 1332

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRDocOnDidInsertPages****Syntax**

```
typedef void (*FRDocOnDidInsertPages)(
    FS_LPVVOID clientData,
    FR_Document doc,
    FS_INT32 iInsertAt,
    FS_INT32 nPages
);
```

**Description**

A callback for document-level event handler. It is called to notify all client that the page has been inserted into the document.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

doc	[In] The <a href="#">FR_Document</a> whose page views has been inserted.
-----	--

---

iInsertAt	[In] The insert point where the page will be inserted.
-----------	--

---

nPages	[In] The page number of inserted page.
--------	--

---

**Return****Head file reference**

fr\_appExpT.h: 1347

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocOnWillDeletePages****Syntax**

```
typedef void (*FRDocOnWillDeletePages)(  
    FS\_LPVVOID clientData,  
    FR\_Document doc,  
    FS\_WordArray arrDelPages  
)
```

**Description**

A callback for document-level event handler. It is called to notify all client that some page views will be deleted from the specified document view. In fact, the [FPD\\_Page](#) is deleted from [FPD\\_Document](#) when this method is calling.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] The <a href="#">FR_Document</a> whose page views will be deleted.
-----	--

---

arrDelPages	[In] The index of the pages that has been deleted.
-------------	--

---

**Return****Head file reference**

fr\_appExpT.h: 1362

**Group**[FR\\_PageEventCallbacksRec](#)**DidDeletePages**

**Syntax**

```
typedef void (*DidDeletePages)(  
    FS\_LPVOID clientData,  
    FR\_Document doc,  
    FS\_WordArray arrDelPages  
)
```

**Description**

A callback for document-level event handler. It is called to notify all client that some page views have been deleted from the specified document view.

**Parameter**

---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> whose page views have been deleted .
arrDelPages	[In] The index of the pages that has been deleted.

---

**Return****Head file reference**

fr\_appExpT.h: 1376

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRDocOnWillModifyPageAttribute****Syntax**

```
typedef void (*FRDocOnWillModifyPageAttribute)(  
    FS\_LPVOID clientData,  
    FR\_Document doc,  
    FS\_INT32 iPage  
)
```

**Description**

A callback for document-level event handler. It is called to notify all client a page view should be reload.

**Parameter**

---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> whose page view's associated <a href="#">FPD_Page</a> will be modified.

---

---

iPage	[In] The index of the page that will be modified.
-------	---

---

**Return****Head file reference**

fr\_appExpT.h: 1390

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocOnDidModifyPageAttribute****Syntax**

```
typedef void (*FRDocOnDidModifyPageAttribute)(  
    FS_LPVVOID clientData,  
    FR_Document doc,  
    FS_INT32 iPage  
)
```

**Description**

A callback for document-level event handler. It is called to notify all client to reload a page.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

doc	[In] The <a href="#">FR_Document</a> whose page view's associated <a href="#">FPD_Page</a> has been modified.
-----	---

---

---

iPage	[In] The index of the page that has been modified.
-------	--

---

**Return****Head file reference**

fr\_appExpT.h: 1404

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocOnWindowCreate****Syntax**

```
typedef void (*FRDocOnWindowCreate)(  
    FS_LPVVOID clientData,  
    FR_Document doc  
)
```

**Description**

A callback for document-level event handler. It is called whenever the doc view is created.

**Parameter**

---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> whose bounding window is being created.

---

**Return****Head file reference**

fr\_appExpT.h: 1417

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRDocOnWindowDestroy****Syntax**

```
typedef void (*FRDocOnWindowDestroy)(  
    FS\_LPVOID clientData,  
    FR\_Document doc  
)
```

**Description**

A callback for document-level event handler. It is called whenever a doc view will be destroyed.

**Parameter**

---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> whose bounding window will be destroyed.

---

**Return****Head file reference**

fr\_appExpT.h: 1430

**Group**

[FR\\_PageEventCallbacksRec](#)

## FRDocOnFrameCreate

### Syntax

```
typedef void (*FRDocOnFrameCreate)(  
    FS_LPVVOID clientData,  
    FR_Document doc,  
    HWND hFrameWnd  
)
```

### Description

A callback for document-level event handler. It is called whenever a child frame window is created.

### Parameter

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> whose parent frame is being created.
hFrameWnd	[In] The child frame window created.

### Return

#### Head file reference

fr\_appExpT.h: 1444

### Group

[FR\\_PageEventCallbacksRec](#)

## FRDocOnFrameDestroy

### Syntax

```
typedef void (*FRDocOnFrameDestroy)(  
    FS_LPVVOID clientData,  
    FR_Document doc,  
    HWND hFrameWnd  
)
```

### Description

A callback for document-level event handler. It is called whenever a child frame window will be destroyed.

### Parameter

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] The <a href="#">FR_Document</a> whose parent frame will be destroyed.
-----	--

---

hFrameWnd	[In] The child frame window to be destroyed.
-----------	--

---

## Return

### Head file reference

fr\_appExpT.h: 1458

### Group

[FR\\_PageEventCallbacksRec](#)

## FRDocOnAnnotSelectionChanged

### Syntax

```
typedef void (*FRDocOnAnnotSelectionChanged)(  
    FS\_LPVVOID clientData  
);
```

### Description

A callback for document-level event handler. It is called whenever a annotation selection changed.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

## Return

### Head file reference

fr\_appExpT.h: 1470

### Group

[FR\\_PageEventCallbacksRec](#)

## FRDocOnAutoScrollBegin

### Syntax

```
typedef void (*FRDocOnAutoScrollBegin)(  
    FS\_LPVVOID clientData,  
    FR\_DocView docView  
);
```

### Description

A callback for document-level event handler. It is called when a document view start to scroll automatically.

**Parameter**


---

clientData	[In] The user-supplied data.
docView	[In] The <a href="#">FR_DocView</a> start to scroll automatically.

---

**Return****Head file reference**

fr\_appExpT.h: 1483

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocOnAutoScrollEnd****Syntax**

```
typedef void (*FRDocOnAutoScrollEnd)(
    FS\_LPVOID clientData,
    FR\_DocView docView
);
```

**Description**

A callback for document-level event handler. It is called whenever an automatic scrolling view stop to scroll.

**Parameter**


---

clientData	[In] The user-supplied data.
docView	[In] The <a href="#">FR_DocView</a> which has been end the automatic scrolling.

---

**Return****Head file reference**

fr\_appExpT.h: 1496

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocOnFinishRender****Syntax**

```
typedef void (*FRDocOnFinishRender)(
    FS\_LPVOID clientData,
    FR\_DocView docView
```

);

**Description**

A callback for document-level event handler.

**Parameter**

---

clientData	[In] The user-supplied data.
docView	[In] The <a href="#">FR_DocView</a> which has been rendered to screen completely.

---

**Return****Head file reference**

fr\_appExpT.h: 1506

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRDocThumbnailWillDraw****Syntax**

```
typedef void (*FRDocThumbnailWillDraw)(  
    FS\_LPVVOID clientData,  
    FR\_ThumbnailView thumbnailView,  
    HDC dc  
)
```

**Description**

A callback for document-level event handler. It is called at the beginning of the drawing.

**Parameter**

---

clientData	[In] The user-supplied data.
thumbnailView	[In] The <a href="#">FR_ThumbnailView</a> that will be drawn.
dc	[In] The device context of <i>thumbnailView</i> .

---

**Return****Head file reference**

fr\_appExpT.h: 1520

**Group**

[FR\\_PageEventCallbacksRec](#)**FRDocThumbnailDidDraw****Syntax**

```
typedef void (*FRDocThumbnailDidDraw)(
    FS_LPVVOID clientData,
    FR_ThumbnailView thumbnailView,
    HDC dc
);
```

**Description**

A callback for document-level event handler. It is called when Reader finished the drawing of a thumbnail view.

**Parameter**

clientData	[In] The user-supplied data.
thumbnailView	[In] The <a href="#">FR_ThumbnailView</a> that will be drawn.
dc	[In] The device context of <i>thumbnailView</i> .

**Return****Head file reference**

fr\_appExpT.h: 1534

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocScrollBarThumbnailViewWillDraw****Syntax**

```
typedef void (*FRDocScrollBarThumbnailViewWillDraw)(
    FS_LPVVOID clientData,
    FR_ScrollBarThumbnailView thumbnailView,
    HDC dc
);
```

**Description**

A callback for document-level event handler. It is called when clicking the scroll bar and Reader beginning the drawing of a thumbnail view.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

thumbnailView	[In] The <a href="#">FR_ScrollBarThumbnailView</a> that will be drawn.
---------------	--

---

dc	[In] The device context of <i>thumbnailView</i> .
----	---

---

**Return****Head file reference**

fr\_appExpT.h: 1548

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocDidFileClose****Syntax**

```
typedef void (*FRDocDidFileClose)(  
    FS\_LPVOID clientData,  
    FS\_LPCWSTR lpwsFilePath  
)
```

**Description**

A callback for document-level event handler. It is called whenever a file is released completely.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

lpwsFilePath	[In] The file path closed.
--------------	----------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 1561

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocCanBeSaved****Syntax**

```
typedef FS_BOOL (*FRDocCanBeSaved)(  
    FS\_LPVOID clientData,  
    FR\_Document doc
```

);

**Description**

A callback for document-level event handler. It is called when user clicked the save button on the save dialog.

**Parameter**

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> that will be saved.

**Return****Head file reference**

fr\_appExpT.h: 1574

**Group**

[FR\\_PageEventCallbacksRec](#)

**OnDocPromptToSave****Syntax**

```
typedef FS_BOOL (*OnDocPromptToSave)(  
    FS_LPVVOID clientData,  
    FR_Document doc,  
    FS_BOOL* bCancel  
)
```

**Description**

A callback for document-level event handler. It is called when the document will be prompted to save.

**Parameter**

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> that will be prompted to save.
bCancel	[Out] Whether to cancel the save process.

**Return****Head file reference**

fr\_appExpT.h: 1588

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocWillReOpen****Syntax**

```
typedef void (*FRDocWillReOpen)(  
    FS_LPVVOID clientData,  
    FR_Document doc,  
    FS_BOOL bMemDoc  
)
```

**Description**

A callback for document-level event handler. It is called by Foxit Reader when the document is to be reopened.

**Parameter**


---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> that will be reopened.
bMemDoc	[In] Indicates whether the document is memory document.

---

**Return****Head file reference**

fr\_appExpT.h: 1602

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocDidReOpen****Syntax**

```
typedef void (*FRDocDidReOpen)(  
    FS_LPVVOID clientData,  
    FR_Document doc,  
    FS_BOOL bMemDoc  
)
```

**Description**

A callback for document-level event handler. It is called by Foxit Reader when the document has been reopened.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] The <a href="#">FR_Document</a> that has been reopened.
-----	--

---

bMemDoc	[In] Indicates whether the document is memory document.
---------	---

---

**Return****Head file reference**

fr\_appExpT.h: 1616

**Group**[FR\\_PageEventCallbacksRec](#)**FRDocOnFrameSize****Syntax**

```
typedef void (*FRDocOnFrameSize)(
    FS\_LPVVOID clientData,
    FR\_Document doc,
    HWND hFrameWnd,
    FS\_Rect rcClient
);
```

**Description**

A callback for document-level event handler. Is is called when the child frame is to be adjusted.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] The <a href="#">FR_Document</a> that is associated with the frame.
-----	---

---

hFrameWnd	[In] The child frame whose size is to be adjusted.
-----------	--

---

rcClient	[In] The client area of the child frame.
----------	--

---

**Return****Head file reference**

fr\_appExpT.h: 1631

**Group**[FR\\_PageEventCallbacksRec](#)

## FRDocOnWillActivate

### Syntax

```
typedef void (*FRDocOnWillActivate)(  
    FS_LPVVOID clientData,  
    FR_Document doc  
)
```

### Description

A callback for document-level event handler. It is called whenever a document is to be activated.

### Parameter

---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> that will become an active document.

---

### Return

#### Head file reference

fr\_appExpT.h: 1644

#### Group

[FR\\_PageEventCallbacksRec](#)

## FRDocOnWillDeactivate

### Syntax

```
typedef void (*FRDocOnWillDeactivate)(  
    FS_LPVVOID clientData,  
    FR_Document doc  
)
```

### Description

A callback for document-level event handler. It is called whenever a document is to be deactivated.

### Parameter

---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> that will become an deactivated document.

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 1658

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRDocOnOtherDocActivate****Syntax**

```
typedef void (*FRDocOnOtherDocActivate)(  
    FS\_LPVOID clientData  
)
```

**Description**

A callback for document-level event handler. It is called whenever a non-PDF document is to be deactivated.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 1672

**Group**

[FR\\_PageEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRDocOnOtherDocDeactivate****Syntax**

```
typedef void (*FRDocOnOtherDocDeactivate)(  
    FS\_LPVOID clientData  
)
```

**Description**

A callback for document-level event handler. It is called whenever a non-PDF document is to be deactivated.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 1686

**Group**[FR\\_PageEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRDocOnOtherDocClose****Syntax**

```
typedef void (*FRDocOnOtherDocClose)(
    FS\_LPVVOID clientData
);
```

**Description**

A callback for document-level event handler. It is called whenever a non-PDF document is to be closed.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 1700

**Group**[FR\\_PageEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRDocDidSave2****Syntax**

```
typedef void (*FRDocDidSave2)(
    FS\_LPVVOID clientData,
    FR\_Document doc,
    FS\_BOOL bSaveAs,
```

```
FS_BOOL bPromptToSave  
);
```

**Description**

A callback for document-level event handler. It is called when a document is saved completely.

**Parameter**

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> that is saved completely.
bSaveAs	[In] A flag indicates whether user click the "save" button or click the "save as" button.
bPromptToSave	[In] Whether Foxit Reader prompts the note to save or not.

**Return****Head file reference**

fr\_appExpT.h: 1716

**Group**

[FR\\_PageEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRDocOnKeyDown****Syntax**

```
typedef FS_BOOL (*FRDocOnKeyDown)(  
    FS_LVOID clientData,  
    FR_DocView docView,  
    unsigned int nKeyCode,  
    unsigned int nFlags  
);
```

**Description**

A callback for document-level event handler. It is called by Foxit Reader to notify the key-down event.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

---

docView	[In] The input doc view whose key-down occurred.
---------	--

---

nKeyCode	[In] The key that was pressed.
----------	--------------------------------

---

nFlags	[In] Indicates whether various virtual keys are down.
--------	---

---

**Return**

[TRUE](#) if the key-down event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 1733

**Group**

[FR\\_PageEventCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 7.3](#)

**FRDocOnKeyUp****Syntax**

```
typedef FS_BOOL (*FRDocOnKeyUp)(  
    FS_LPVVOID clientData,  
    FR_DocView docView,  
    unsigned int nKeyCode,  
    unsigned int nFlags  
)
```

**Description**

A callback for document-level event handler. It is called by Foxit Reader to notify the key-up event.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

docView	[In] The input doc view whose key-up occurred.
---------	--

---

nKeyCode	[In] The key that was pressed.
----------	--------------------------------

---

nFlags	[In] Indicates whether various virtual keys are down.
--------	---

---

**Return**

[TRUE](#) if the key-up event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 1750

**Group**[FR\\_PageEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.3](#)**FRDocOnChar****Syntax**

```
typedef FS_BOOL (*FRDocOnChar)(
    FS_LPVVOID clientData,
    FR_DocView docView,
    unsigned int nKeyCode,
    unsigned int nFlags
);
```

**Description**

A callback for document-level event handler. It is called by Foxit Reader to notify the char-click event.

**Parameter**

clientData	[In] The user-supplied data.
docView	[In] The input doc view whose char-click occurred.
nKeyCode	[In] The char that was pressed.
nFlags	[In] Indicates whether various virtual keys are down.

**Return**

[TRUE](#) if the char-click event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 1767

**Group**[FR\\_PageEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.3](#)**FRDocOnLButtonDown**

**Syntax**

```
typedef FS_BOOL (*FRDocOnLButtonDown)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    unsigned int nFlags,  
    FS_DevicePoint point  
)
```

**Description**

A callback for document-level event handler. It is called by Foxit Reader to notify the left-button-down event.

**Parameter**


---

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose left-button-down event occurred.
nFlags	[In] Indicates whether various virtual keys are down.
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

---

**Return**

[TRUE](#) if the left-button-down event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 1784

**Group**

[FR\\_PageEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FRDocOnLButtonUp****Syntax**

```
typedef FS_BOOL (*FRDocOnLButtonUp)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    unsigned int nFlags,  
    FS_DevicePoint point  
)
```

**Description**

A callback for document-level event handler. It is called by Foxit Reader to notify the left-button-up event.

#### Parameter

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose left-button-up event occurred.
nFlags	[In] Indicates whether various virtual keys are down.
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

#### Return

[TRUE](#) if the left-button-up event was handled, otherwise not.

#### Head file reference

fr\_appExpT.h: 1801

#### Group

[FR\\_PageEventCallbacksRec](#)

#### Since

[SDK LATEEST VERSION > 7.3](#)

## FRDocOnLButtonDblClk

#### Syntax

```
typedef FS_BOOL (*FRDocOnLButtonDblClk)(  
    FS_LVOID clientData,  
    FR_PageView pageView,  
    unsigned int nFlags,  
    FS_DevicePoint point  
)
```

#### Description

A callback for document-level event handler. It is called by Foxit Reader to notify the left-button-double-click event.

#### Parameter

clientData	[In] The user-supplied data.
------------	------------------------------

---

pageView	[In] The input page view whose left-button-double-click event occurred.
----------	---

---

nFlags	[In] Indicates whether various virtual keys are down.
--------	---

---

point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.
-------	---

---

**Return**

[TRUE](#) if the left-button-double-click event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 1818

**Group**

[FR\\_PageEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FRDocOnMouseMove****Syntax**

```
typedef FS_BOOL (*FRDocOnMouseMove)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    unsigned int nFlags,  
    FS_DevicePoint point  
)
```

**Description**

A callback for document-level event handler. It is called by Foxit Reader to notify the mouse-move event.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

pageView	[In] The input page view whose mouse-move event occurred.
----------	---

---

nFlags	[In] Indicates whether various virtual keys are down.
--------	---

---

point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.
-------	---

---

**Return**

TRUE if the mouse-move event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 1835

**Group**

[FR\\_PageEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FRDocOnRButtonDown****Syntax**

```
typedef FS_BOOL (*FRDocOnRButtonDown)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    unsigned int nFlags,  
    FS_DevicePoint point  
)
```

**Description**

A callback for document-level event handler. It is called by Foxit Reader to notify the right-button-down event.

**Parameter**


---

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose right-button-down event occurred.
nFlags	[In] Indicates whether various virtual keys are down.
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

---

**Return**

TRUE if the right-button-down event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 1852

**Group**

[FR\\_PageEventCallbacksRec](#)

**Since**[SDK\\_LATEEST\\_VERSION > 7.3](#)

## FRDocOnRButtonUp

**Syntax**

```
typedef FS_BOOL (*FRDocOnRButtonUp)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,  
    unsigned int nFlags,  
    FS_DevicePoint point  
)
```

**Description**

A callback for document-level event handler. It is called by Foxit Reader to notify the right-button-up event.

**Parameter**


---

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose right-button-up event occurred.
nFlags	[In] Indicates whether various virtual keys are down.
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

---

**Return**

[TRUE](#) if the right-button-up event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 1869

**Group**

[FR\\_PageEventCallbacksRec](#)

**Since**[SDK\\_LATEEST\\_VERSION > 7.3](#)

## FRDocOnRButtonDblClk

**Syntax**

```
typedef FS_BOOL (*FRDocOnRButtonDblClk)(  
    FS_LPVVOID clientData,  
    FR_PageView pageView,
```

---

```
unsigned int nFlags,
FS\_DevicePoint point
);
```

**Description**

A callback for document-level event handler. It is called by Foxit Reader to notify the right-button-double-click event.

**Parameter**

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose right-button-double-click event occurred.
nFlags	[In] Indicates whether various virtual keys are down.
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

**Return**

[TRUE](#) if the right-button-double-click event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 1886

**Group**

[FR\\_PageEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FRDocOnMouseWheel****Syntax**

```
typedef FS_BOOL (*FRDocOnMouseWheel)(
    FS\_LVOID clientData,
    FR\_PageView pageView,
    unsigned int nFlags,
    short zDelta,
    FS\_DevicePoint point
);
```

**Description**

A callback for document-level event handler. It is called by Foxit Reader to notify the mouse-wheel event.

**Parameter**


---

clientData	[In] The user-supplied data.
pageView	[In] The input page view whose mouse-wheel event occurred.
nFlags	[In] Indicates whether various virtual keys are down.
zDelta	[In] Indicates distance rotated. Reference to <i>CWnd::OnMouseWheel</i> .
point	[In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

---

**Return**

[TRUE](#) if the mouse-wheel event was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 1904

**Group**

[FR\\_PageEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FRDocOnDrawAnnot****Syntax**

```
typedef void (*FRDocOnDrawAnnot)(
    FS_LPVVOID clientData,
    FR_PageView pageView,
    FR_Annot frAnnot,
    HDC hDC,
    FPD_RenderDevice renderDevice,
    FS_AffineMatrix user2Device,
    FS_Rect rcWindow
);
```

**Description**

A callback for document-level event handler. It is called by Foxit Reader when the specified annotation has been drawn.

**Parameter**

---

clientData	[In] The user-supplied data.
pageView	[In] The input page view where the specified annotation is drawn.
frAnnot	[In] The specified annotation to be drawn.
hDC	[In] The device context of <i>pageView</i> .
renderDevice	[In] The memory device used to render the annotation.
user2Device	[In] The current displaying transformation matrix.
rcWindow	[In] The rectangle of <i>pageView</i> .

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 1924

**Group**[FR\\_PageEventCallbacksRec](#)**Since**[SDK LATEEST VERSION > 7.3](#)**FRDocOnDocCollectActionData****Syntax**

```
typedef void (*FRDocOnDocCollectActionData)(
    FS_LPVVOID clientData,
    FS_Document doc,
    FS_LPCWSTR lpwsOperatorType,
    FS_LPCWSTR lpwsOperator,
    FS_MapPtrToPtr valueMap
);
```

**Description**

A callback for document-level event handler. It is called to notify the plug-in that some actions occur.

**Parameter**

---

clientData	[In] The user-supplied data.
doc	[In] The specified <a href="#">FR Document</a> .
lpwsOperatorType	[In] The operator type.
lpwsOperator	[In] The operator.
valueMap	[In] The count of the content related to the operands.

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 1942

**Group**[FR\\_PageEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.2.2](#)**FRDocWillOpen2****Syntax**

```
typedef void (*FRDocWillOpen2)(
    FS\_LPVOID clientData,
    FR Document doc,
    FS LPCWSTR lpwsFilePath
);
```

**Description**

A callback for document-level event handler. It is called whenever a document is opening. You can uses this new interface instead of [FRDocWillOpen](#). If [FRDocWillOpen](#) has been set, [FRDocWillOpen2](#) will be ignored.

**Parameter**


---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR Document</a> object that is opening.
lpwsFilePath	[In] The file path to be opened.

---

**Return****Head file reference**

fr\_appExpT.h: 1958

**Group**[FR\\_PageEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.3](#)**FRDocOnOptimizerFinish****Syntax**

```
typedef void (*FRDocOnOptimizerFinish)(  
    FS\_LPVOID clientData,  
    FR\_Document doc  
)
```

**Description**

A callback for document-level event handler. It is called whenever a document has been optimized.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] The <a href="#">FR_Document</a> object that is optimized.
-----	--

---

**Return****Head file reference**

fr\_appExpT.h: 1973

**Group**[FR\\_PageEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.3.1](#)**FRDocCanBeClose****Syntax**

```
typedef FS_BOOL (*FRDocCanBeClose)(  
    FS\_LPVOID clientData,  
    FR\_Document doc  
)
```

**Description**

A callback for document-level event handler. It is called whenever a document is to be closed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] The <a href="#">FR_Document</a> object that is closed.
-----	---

---

**Return****Head file reference**

fr\_appExpT.h: 1987

**Group**

[FR\\_PageEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRDocOnReOpenFailed****Syntax**

```
typedef void (*FRDocOnReOpenFailed)(  
    FS\_LPVOID clientData,  
    FS LPCWSTR lpwsFilePath  
);
```

**Description**

A callback for document-level event handler. It is called whenever the document cannot be reopened successfully.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

lpwsFilePath	[In] The file path closed.
--------------	----------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 2001

**Group**

[FR\\_PageEventCallbacksRec](#)

**Since**  
[SDK LATEEST VERSION > 8.1](#)

## FRDocWillSave2

### Syntax

```
typedef FS_BOOL (*FRDocWillSave2)(  
    FS_LPVVOID clientData,  
    FR_Document doc,  
    FS_BOOL bSaveAs,  
    FS_LPCWSTR lpwsFilePath,  
    clientData,  
    FS_BOOL bCanSupportPDFOnly,  
    FS_BOOL bChoise,  
    FS_LPCWSTR* pwszFilePath,  
    FS_INT32 iIndex  
)
```

### Description

A callback for document-level event handler. [SDK LATEEST VERSION](#) > 8.2It is called when user clicked the save button on the save dialog before Show Cfiledialog.

### Parameter

---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> that will be saved.
bSaveAs	[In] A flag indicates whether user click the "save" button or click the "save as" button.
lpwsFilePath	[In] The file path of the document that will be saved.
clientData	[In] The user-supplied data.
bCanSupportPDFOnly	[In] SaveAs type only can be choose pdf type
bChoise	[Out] the value of user click ok or cancel button
pwszFilePath	[Out] The SaveAs file path name
iIndex	[Out] The SaveAs type index

---

### Return

**Head file reference**

fr\_appExpT.h: 2033

**Group**[FR\\_PageEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 8.3](#)**FRDocSaveAsBeforeReopen****Syntax**

```
typedef FS_BOOL (*FRDocSaveAsBeforeReopen)(  
    FS_LPVOID clientData,  
    FS_LPCWSTR wsFileName  
)
```

**Description**

A callback for document-level event handler. It is called After SaveAs before Reopen.

**Parameter**

---

clientData	[In] The user-supplied data.
wsFileName	[In] The file path name.

---

**Return****Head file reference**

fr\_appExpT.h: 2051

**Group**[FR\\_PageEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 8.3](#)**FRDocCanPaste****Syntax**

```
typedef FS_BOOL (*FRDocCanPaste)(  
    FS_LPVOID clientData  
)
```

**Description**

A callback for document-level event handler. It is called When Paste the data to determine whether the current data can be pasted

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

### Return

#### Head file reference

fr\_appExpT.h: 2064

#### Group

[FR\\_PageEventCallbacksRec](#)

#### Since

[SDK\\_LATEEST\\_VERSION > 8.3](#)

## OnMouseClickOnText

### Syntax

```
typedef void (*OnMouseClickOnText)(  
    FS_LPVOID clientData,  
    FR_Document doc,  
    FS_LPCWSTR wsText,  
    FS_Rect rect  
)
```

### Description

A callback for document-level event handler. It is called when the mouse clicks on the text field.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] It identifies the document opened. It can be used as a unique identifier for the document.
-----	---

---

wsText	[In] The text where the mouse clicks on.
--------	--

---

rect	[In] The text rectangle where the mouse clicks on.
------	--

---

### Return

#### Head file reference

fr\_appExpT.h: 2080

**Group**[FR\\_PageEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 8.3.2](#)**FRDocOwnerSaveAs****Syntax**

```
typedef BOOL (*FRDocOwnerSaveAs)(
    FS\_LPVOID clientData,
    FR\_Document doc,
    FS\_LPCWSTR wszPathName
);
```

**Description**

A callback for document-level event handler. It is called after click the ok of saveAs dialog, before save operation.

**Parameter**

clientData	[In] The user-supplied data.
doc	[In] It identifies the document saved. It can be used as a unique identifier for the document.
wszPathName	[In] The path of the save document.

**Return****Head file reference**

fr\_appExpT.h: 2095

**Group**[FR\\_PageEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 9.1](#)**FRDocOnCanDetache****Syntax**

```
typedef FS_BOOL (*FRDocOnCanDetache)(
    FS\_LPVOID clientData,
    FR\_Document doc
);
```

**Description**

A callback for document-level event handler. It is called whenever a document can drop to other frame.

**Parameter**

---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR Document</a> that will become an active document.

---

**Return****Head file reference**

fr\_appExpT.h: 2108

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRDocDelayDidOpen****Syntax**

```
typedef void (*FRDocDelayDidOpen)(  
    FR Document doc  
)
```

**Description**

A callback for document-level event handler. It is called whenever a document have opened.

**Parameter**

---

doc	[In] The <a href="#">FR Document</a> object that is optimized.
-----	--

---

**Return****Head file reference**

fr\_appExpT.h: 2121

**Group**

[FR\\_PageEventCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 7.3.1](#)

**FRDocOnActivate2**

**Syntax**

```
typedef void (*FRDocOnActivate2)(
    FS\_LPVVOID clientData,
    FR\_Document doc,
    FS\_BOOL bMainfrmActivating
);
```

**Description**

A callback for document-level event handler. It is called whenever a document is to be active.

**Parameter**

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> that becomes an active document.
bMainfrmActivating	[In] If document activate in function OnActivate or OnActivateTopLevel.

**Return****Head file reference**

fr\_appExpT.h: 2138

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRDocOnDeactivate2****Syntax**

```
typedef void (*FRDocOnDeactivate2)(
    FS\_LPVVOID clientData,
    FR\_Document doc,
    FS\_BOOL bMainfrmActivating,
    clientData,
    bMainfrmActivating,
    clientData,
    doc,
    FR\_Annot focusAnnot,
    clientData,
    FR\_PageView pv
);
```

**Description**

A callback for page-level event handler. [SDK\\_LATEEST\\_VERSION](#) > 9.5It is called whenever a page of a PDF document will be opened.

**Parameter**

---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> that becomes an deactivate document.
bMainfrmActivating	[In] If document deactivate in function OnActivate or OnActivateTopLevel.
clientData	[In] The user-supplied data.
bMainfrmActivating	[In] If other document activate in function OnActivate or OnActivateTopLevel.
clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FR_Document</a> where the annotation is located.
focusAnnot	[In] The <a href="#">FR_Annot</a> focused.
clientData	[In] The user-supplied data.
pv	[In] The <a href="#">FR_PageView</a> which is to be opened.

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 2211

**Group**[FR\\_PageEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 9.5](#)**FRPageViewOnClose****Syntax**

```
typedef void (*FRPageViewOnClose)(
    FS\_LPVVOID clientData,
    FR\_PageView pv
);
```

**Description**

A callback for page-level event handler. It is called whenever a page of a document will be closed.

**Parameter**

---

clientData	[In] The user-supplied data.
pv	[In] The <a href="#">FR_PageView</a> which is to be closed.

---

**Return****Head file reference**

fr\_appExpT.h: 2224

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRPageViewOnVisible****Syntax**

```
typedef void (*FRPageViewOnVisible)(  
    FS\_LPVOID clientData,  
    FR\_PageView pv  
);
```

**Description**

A callback for page-level event handler. It is called whenever a page will be visible.

**Parameter**

---

clientData	[In] The user-supplied data.
pv	[In] The <a href="#">FR_PageView</a> which will be visible.

---

**Return****Head file reference**

fr\_appExpT.h: 2238

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRPageViewInvisible****Syntax**

```
typedef void (*FRPageViewInvisible)(  
    FS_LPVVOID clientData,  
    FR_PageView pv  
>;
```

**Description**

A callback for page-level event handler. It is called whenever a page is going to be invisible.

**Parameter**

clientData	[In] The user-supplied data.
pv	[In] The <a href="#">FR_PageView</a> which will be invisible.

**Return****Head file reference**

fr\_appExpT.h: 2252

**Group**

[FR\\_PageEventCallbacksRec](#)

**FRPageViewOnContentChanged****Syntax**

```
typedef void (*FRPageViewOnContentChanged)(  
    FS_LPVVOID clientData,  
    FR_PageView pv  
>;
```

**Description**

A callback for page-level event handler. It is called whenever a page's content is modified.

**Parameter**

clientData	[In] The user-supplied data.
pv	[In] The <a href="#">FR_PageView</a> whose content stream has been modified.

**Return****Head file reference**

fr\_appExpT.h: 2265

**Group**

[FR\\_PageEventCallbacksRec](#)**FRPageViewOnWillParsePage****Syntax**

```
typedef void (*FRPageViewOnWillParsePage)(  
    FS_LPVVOID clientData,  
    FR_PageView pv,  
    FS_BOOL bPageVisible  
)
```

**Description**

A callback for page-level event handler. It is called when a page is going to be parsed.

**Parameter**


---

clientData	[In] The user-supplied data.
pv	[In] The <a href="#">FR_PageView</a> whose page is going to be parsed.
bPageVisible	[In] Indicates whether the page is visible

---

**Return****Head file reference**

fr\_appExpT.h: 2280

**Group**[FR\\_PageEventCallbacksRec](#)**Since**

[SDK LATEST VERSION > 2.1](#)

**FRPageViewOnDidParsePage****Syntax**

```
typedef void (*FRPageViewOnDidParsePage)(  
    FS_LPVVOID clientData,  
    FR_PageView pv,  
    FS_BOOL bPageVisible  
)
```

**Description**

A callback for page-level event handler. It is called when a page has been parsed.

**Parameter**

---

clientData	[In] The user-supplied data.
pv	[In] The <a href="#">FR_PageView</a> whose page has been parsed.
bPageVisible	[In] Indicates whether the page is visible

---

**Return****Head file reference**

fr\_appExpT.h: 2295

**Group**[FR\\_PageEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.1](#)**FRPageViewOnContentChanged2****Syntax**

```
typedef void (*FRPageViewOnContentChanged2)(  
    FS\_LPVOID clientData,  
    FR\_PageView pv,  
    FS\_PtrArray objArray,  
    FR\_ContentChangeType changeType  
)
```

**Description**

A callback for page-level event handler. It is called whenever a page's content is modified.

**Parameter**


---

clientData	[In] The user-supplied data.
pv	[In] The <a href="#">FR_PageView</a> whose content stream has been modified.
objArray	[In] It indicates what data is changed. The value type in the array is <a href="#">FR_ChangedContent</a> .
changeType	[In] The content change type.

---

**Return****Head file reference**

fr\_appExpT.h: 2311

**Group**

[FR\\_PageEventCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 9.1](#)

## FRPanelViewGetName

**Syntax**

```
typedef FS_LPSTR (*FRPanelViewGetName)(  
    FS_LPVVOID clientData  
)
```

**Description**

A callback for navigation panel view. It is called by Foxit Reader to get the panel view's name.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

A null-determined string correspond the name of panel view.

**Head file reference**

fr\_appExpT.h: 3430

**Group**

[FR\\_PanelViewCallbacksRec](#)

## FRPanelViewGetTitle

**Syntax**

```
typedef FS_LPWSTR (*FRPanelViewGetTitle)(  
    FS_LPVVOID clientData  
)
```

**Description**

A callback for navigation panel view. It is called by Foxit Reader to get the panel view's title which is shown in user interface.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

A null-determined string correspond the title of panel view.

**Head file reference**

fr\_appExpT.h: 3441

**Group**

[FR\\_PanelViewCallbacksRec](#)

**FRPanelViewInitNewView****Syntax**

```
typedef void (*FRPanelViewInitNewView)(  
    FS\_LPVVOID clientData,  
    FPD\_Document doc,  
    HWND window  
);
```

**Description**

A callback for navigation panel view. It is called when a PDF document is open and to be displayed. The plug-in can init thedata of the panel view in this callback. Then create the window attached to the panel view in [FRPanelViewOnPanelActive](#) .

**Parameter**

---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FPD_Document</a> associated with the created window.
window	[In] The parent window.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 3454

**Group**

[FR\\_PanelViewCallbacksRec](#)

**FRPanelViewOnPanelActive****Syntax**

```
typedef HWND (*FRPanelViewOnPanelActive)(  
    FS\_LPVVOID clientData  
);
```

**Description**

A callback for navigation panel view. It creates a window to attach to navigation panel. Using the window API function *CreateWindow* or *CreateWindowEx* to create a window for returning.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

A new window.

**Head file reference**

fr\_appExpT.h: 3466

**Group**

[FR\\_PanelViewCallbacksRec](#)

**FRPanelViewOnActive****Syntax**

```
typedef void (*FRPanelViewOnActive)(  
    FS\_LPVOID clientData,  
    FPD\_Document doc,  
    HWND window  
)
```

**Description**

A callback for navigation panel view. It is called when a panel view becomes a active view.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

doc	[In] The <a href="#">FPD_Document</a> associated with the active view.
-----	--

---

---

window	[In] The parent window.
--------	-------------------------

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 3478

**Group**

[FR\\_PanelViewCallbacksRec](#)**FRPanelViewOnRotate****Syntax**

```
typedef void (*FRPanelViewOnRotate)(  
    FS_LPVVOID clientData,  
    FPD_Document doc,  
    FS_INT32 nRotate  
)
```

**Description**

A callback for navigation panel view to rotate. It is called after the page view is rotated. You should do some rotation with the panel view.

**Parameter**


---

clientData	[In] The user-supplied data.
doc	[In] The <a href="#">FPD_Document</a> associated with the created window.
nRotate	[In] The rotation flag. See <a href="#">FR_RotationFlags</a> group.

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 3491

**Group**[FR\\_PanelViewCallbacksRec](#)**FRPanelViewOnDestroyView****Syntax**

```
typedef void (*FRPanelViewOnDestroyView)(  
    FS_LPVVOID clientData,  
    HWND window,  
    FPD_Document doc  
)
```

**Description**

A callback for navigation panel view. It is called at the shutdown time of panel view to free dynamic memory. The panel view should be destroyed in this method.

**Parameter**

---

clientData	[In] The user-supplied data.
window	[In] The panel view to destroy.
doc	[In] The <a href="#">FPD Document</a> associated with current document.

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 3504

**Group**[FR\\_PanelViewCallbacksRec](#)**FRPanelViewIsDockToBottom****Syntax**

```
typedef FS_BOOL (*FRPanelViewIsDockToBottom)(
    FS\_LVOID clientData
);
```

**Description**

A callback for navigation panel view. It is called to tell Foxit Reader whether the panel view need to dock bottom.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

if [TRUE](#), Foxit Reader will dock the panel view at the bottom, otherwise at the left.

**Head file reference**

fr\_appExpT.h: 3515

**Group**[FR\\_PanelViewCallbacksRec](#)**FRPanelViewGetButtonTip****Syntax**

```
typedef void (*FRPanelViewGetButtonTip)(
    FS\_LVOID clientData,
    FS\_WideString* csOutTip
);
```

**Description**

A callback for navigation panel view to pass the tip of panel button. Gets the tip of panel button that displaying on the left bar of panel.

**Parameter**

---

clientData	[In] The user-supplied data.
csOutTip	[In] (Passed by reader, filled by this method) The buffer to fill with the tip.

---

**Return****Head file reference**

fr\_appExpT.h: 3526

**Group**

[FR\\_PanelViewCallbacksRec](#)

**FRPanelViewGetButtonDescriptText****Syntax**

```
typedef void (*FRPanelViewGetButtonDescriptText)(  
    FS_LPVOID clientData,  
    FS_WideString* csOutText  
)
```

**Description**

A callback for navigation panel view to tell the describe text. Gets the describe text of panel view. The describe text will be displayed on the left corner of status bar when the mouse over the panel button. This interface is not available in version 1.0.

**Parameter**

---

clientData	[In] The user-supplied data.
csOutText	[In] (Passed by reader, filled by this method) The buffer to fill with the text.

---

**Return****Head file reference**

fr\_appExpT.h: 3539

**Group**

[FR\\_PanelViewCallbacksRec](#)

## FRPanelViewGetButtonIcon

### Syntax

```
typedef FS_DIBitmap (*FRPanelViewGetButtonIcon)(  
    FS_LPVVOID clientData  
);
```

### Description

A callback for navigation panel view to get the icon of panel button. The bitmap returned by this method will be taken by Reader, client can not release it until user exit Reader.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

### Return

A [FS\\_DIBitmap](#). The size of bitmap is 24 \* 24.

### Head file reference

fr\_appExpT.h: 3551

### Group

[FR\\_PanelViewCallbacksRec](#)

## FRPanelViewSetPos

### Syntax

```
typedef void (*FRPanelViewSetPos)(  
    FS_LPVVOID clientData,  
    FPD_Document doc,  
    FS_INT32 nPage  
);
```

### Description

A callback for navigation panel view to locate the navigation item(such as bookmark item). It is called when user scroll a page view or go to other pages.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] The <a href="#">FPD_Document</a> associated with the created window.
-----	---

---

nPage	[In] The index of current page.
-------	---------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 3565

**Group**[FR\\_PanelViewCallbacksRec](#)**FRPDFA\_SaveAsPDFA****Syntax**

```
typedef FS_BOOL (*FRPDFA_SaveAsPDFA)(  
    FS_LPVVOID clientData,  
    FR_Document pRDoc,  
    const FRPDFA_PDFVersion pVersion,  
    FS_WideString wsPathSuffix  
) ;
```

**Description****Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

pRDoc	[In]
-------	------

---

pVersion	[In]
----------	------

---

wsPathSuffix	[In]
--------------	------

---

**Return****Head file reference**

fr\_appExpT.h: 6106

**Group**[FR\\_PDFAPrinterCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 9.1](#)**FRPOOnBeforeInsertPages****Syntax**

```
typedef void (*FRPOOnBeforeInsertPages)()
```

---

```
FS_LPVVOID clientData,
FR_Document frDoc,
FS_INT32 nInsertAt,
FS_INT32 nCount
);
```

**Description**

A callback for page organizing event handler. It is called before the Foxit Reader inserts the pdf pages.

**Parameter**

clientData	[In] The user-supplied data.
frDoc	[In] The document where the pdf pages are inserted to.
nInsertAt	[In] The index where the pdf pages are inserted at.
nCount	[In] The count the pdf pages are inserted.

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5568

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRPOOnDoInsertPagesDictFinish****Syntax**

```
typedef void (*FRPOOnDoInsertPagesDictFinish)(
    FS_LPVVOID clientData,
    FPD_Document fpdDesDoc,
    FS_INT32 nInsertAt,
    FPD_Document fpdSrcDoc,
    FS_WordArray arrSrcPages,
    FS_BOOL bEntireDoc
);
```

**Description**

A callback for page organizing event handler. It is called when the Foxit Reader finishes to insert the dictionaries to the pdf pages.

**Parameter**


---

clientData	[In] The user-supplied data.
fpdDesDoc	[In] The dest document.
nInsertAt	[In] The index where the pdf pages are inserted at.
fpdSrcDoc	[In] The source document.
arrSrcPages	[In] The array to store the source pages
bEntireDoc	[In] It indicates whether the operation is on the entire document.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5587

**Group**[FR\\_POEventCallbacksRec](#)**Since**[SDK LATEEST VERSION > 2.1.0.4](#)**FRPOOnAfterInsertPages****Syntax**

```
typedef void (*FRPOOnAfterInsertPages)(
    FS\_LPVVOID clientData,
    FR\_Document frDoc,
    FS\_INT32 nInsertAt,
    FS\_INT32 nCount
);
```

**Description**

A callback for page organizing event handler. It is called after the Foxit Reader inserts the pdf pages.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

frDoc	[In] The document where the pdf pages are inserted to.
-------	--

---

nInsertAt	[In] The index where the pdf pages are inserted at.
-----------	---

---

nCount	[In] The count the pdf pages are inserted.
--------	--

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5604

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FRPOOnBeforeDeletePages****Syntax**

```
typedef void (*FRPOOnBeforeDeletePages)(  
    FS\_LPVVOID clientData,  
    FR\_Document frDoc,  
    FS\_WordArray arrDelPages  
)
```

**Description**

A callback for page organizing event handler. It is called before the Foxit Reader deletes the pdf pages.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The document where the pdf pages are deleted from.
-------	---

---

arrDelPages	[In] The index of the pages that has been deleted.
-------------	--

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5620

**Group**

[FR\\_POEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)**FRPOOnAfterDeletePages****Syntax**

```
typedef void (*FRPOOnAfterDeletePages)(  
    FS\_LPVVOID clientData,  
    FR\_Document frDoc,  
    FS\_WordArray arrDelPages  
)
```

**Description**

A callback for page organizing event handler. It is called after the Foxit Reader deletes the pdf pages.

**Parameter**

---

clientData	[In] The user-supplied data.
frDoc	[In] The document where the pdf pages are deleted from.
arrDelPages	[In] The index of the pages that has been deleted.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5636

**Group**[FR\\_POEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)**FRPOOnBeforeReplacePages****Syntax**

```
typedef void (*FRPOOnBeforeReplacePages)(  
    FS\_LPVVOID clientData,  
    FR\_Document frDoc,  
    FS\_INT32 nStart,  
    FPD\_Document fpdSrcDoc,  
    FS\_WordArray arrSrcPages  
)
```

**Description**

A callback for page organizing event handler. It is called before the Foxit Reader replaces the pdf pages.

**Parameter**


---

clientData	[In] The user-supplied data.
frDoc	[In] The document where the pdf pages are replaced.
nStart	[In] The start index where the pdf pages are replaced from.
fpdSrcDoc	[In] The source document.
arrSrcPages	[In] The array to store the source pages.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5654

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRPOOnAfterReplacePages****Syntax**

```
typedef void (*FRPOOnAfterReplacePages)(
    FS_LPVOID clientData,
    FR_Document frDoc,
    FS_INT32 nStart,
    FPD_Document fpdSrcDoc,
    FS_WordArray arrSrcPages
);
```

**Description**

A callback for page organizing event handler. It is called after the Foxit Reader replaces the pdf pages.

**Parameter**

---

clientData	[In] The user-supplied data.
frDoc	[In] The document where the pdf pages are replaced.
nStart	[In] The start index where the pdf pages are replaced from.
fpdSrcDoc	[In] The source document.

---

arrSrcPages	[In] The array to store the source pages.
-------------	---

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5672

**Group**[FR\\_POEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)**FRPOOnBeforeSwapTwoPage****Syntax**

```
typedef void (*FRPOOnBeforeSwapTwoPage)(
    FS\_LPVOID clientData,
    FR\_Document frDoc,
    FS\_INT32 iPage1,
    FS\_INT32 iPage2
);
```

**Description**

A callback for page organizing event handler. It is called before the Foxit Reader swaps two pdf pages.

**Parameter**


---

clientData	[In] The user-supplied data.
frDoc	[In] The document where the pdf pages are swapped.
iPage1	[In] The first page index.

---

---

iPage2	[In] The second page index.
--------	-----------------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5689

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FRPOOnAfterSwapTwoPage****Syntax**

```
typedef void (*FRPOOnAfterSwapTwoPage)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 iPage1,  
    FS_INT32 iPage2  
)
```

**Description**

A callback for page organizing event handler. It is called after the Foxit Reader swaps two pdf pages.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

frDoc	[In] The document where the pdf pages are swapped.
-------	--

---

---

iPage1	[In] The first page index.
--------	----------------------------

---

---

iPage2	[In] The second page index.
--------	-----------------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5706

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

## FRPOOnBeforeRotatePage

**Syntax**

```
typedef void (*FRPOOnBeforeRotatePage)(  
    FS\_LPVOID clientData,  
    FR\_Document frDoc,  
    FS\_INT32 iPage,  
    FS\_INT32 nRotate  
)
```

**Description**

A callback for page organizing event handler. It is called before the Foxit Reader rotates the pdf page.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The document where the pdf page is rotated.
-------	--

---

iPage	[In] The page index.
-------	----------------------

---

nRotate	[In] The rotation value.
---------	--------------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5723

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

## FRPOOnAfterRotatePage

**Syntax**

```
typedef void (*FRPOOnAfterRotatePage)(  
    FS\_LPVOID clientData,  
    FR\_Document frDoc,  
    FS\_INT32 iPage,  
    FS\_INT32 nRotate
```

);

**Description**

A callback for page organizing event handler. It is called after the Foxit Reader rotates the pdf page.

**Parameter**

clientData	[In] The user-supplied data.
frDoc	[In] The document where the pdf page is rotated.
iPage	[In] The page index.
nRotate	[In] The rotation value.

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5740

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRPOOnBeforeResizePage****Syntax**

```
typedef void (*FRPOOnBeforeResizePage)(  
    FS\_LPVOID clientData,  
    FR\_Document frDoc,  
    FS\_INT32 iPage,  
    FS\_FloatRect MediaBox,  
    FS\_FloatRect CropBox  
)
```

**Description**

A callback for page organizing event handler. It is called before the Foxit Reader resizes the pdf page.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The document where the pdf page is resized.
-------	--

---

iPage	[In] The page index.
-------	----------------------

---

MediaBox	[In] The media box.
----------	---------------------

---

CropBox	[In] The crop box.
---------	--------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5758

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FRPOOnAfterResizePage****Syntax**

```
typedef void (*FRPOOnAfterResizePage)(  
    FS\_LPVVOID clientData,  
    FR\_Document frDoc,  
    FS\_INT32 iPage,  
    FS\_FloatRect MediaBox,  
    FS\_FloatRect CropBox  
)
```

**Description**

A callback for page organizing event handler. It is called after the Foxit Reader resizes the pdf page.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The document where the pdf page is resized.
-------	--

---

iPage	[In] The page index.
-------	----------------------

---

MediaBox	[In] The media box.
----------	---------------------

---

---

CropBox	[In] The crop box.
---------	--------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5776

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FRPOOnBeforeExtractPage****Syntax**

```
typedef void (*FRPOOnBeforeExtractPage)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc,  
    FS_WordArray arrSrcPages,  
    FPD_Document fpdDstDoc  
)
```

**Description**

A callback for page organizing event handler. It is called before the Foxit Reader extracts the pdf pages.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

frDoc	[In] The document where the pdf pages are extracted.
-------	--

---

---

arrSrcPages	[In] The array to store the source pages.
-------------	---

---

---

fpdDstDoc	[In] The dest document.
-----------	-------------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5793

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FRPOOnAfterExtractPage****Syntax**

```
typedef void (*FRPOOnAfterExtractPage)(  
    FS\_LPVVOID clientData,  
    FR\_Document frDoc,  
    FS\_WordArray arrSrcPages,  
    FPD\_Document fpdDstDoc  
)
```

**Description**

A callback for page organizing event handler. It is called after the Foxit Reader extracts the pdf pages.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The document where the pdf pages are extracted.
-------	--

---

arrSrcPages	[In] The array to store the source pages.
-------------	---

---

fpdDstDoc	[In] The dest document.
-----------	-------------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5810

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FRPOOnBeforeModifyPageAttr****Syntax**

```
typedef void (*FRPOOnBeforeModifyPageAttr)(  
    FS\_LPVVOID clientData,  
    FR\_Document frDoc,  
    FS\_INT32 nStart,
```

---

```
FS_INT32 nCount
);
```

**Description**

A callback for page organizing event handler. It is called before the Foxit Reader modifies the attributes of the pdf pages.

**Parameter**


---

clientData	[In] The user-supplied data.
frDoc	[In] The document where the attributes of the pdf pages are modified.
nStart	[In] The start index.
nCount	[In] The page count.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5827

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FRPOOnAfterModifyPageAttr****Syntax**

```
typedef void (*FRPOOnAfterModifyPageAttr)(
    FS_LPVVOID clientData,
    FR_Document frDoc,
    FS_INT32 nStart,
    FS_INT32 nCount
);
```

**Description**

A callback for page organizing event handler. It is called after the Foxit Reader modifies the attributes of the pdf pages.

**Parameter**

---

clientData	[In] The user-supplied data.
frDoc	[In] The document where the attributes of the pdf pages are modified.
nStart	[In] The start index.
nCount	[In] The page count.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5844

**Group**[FR\\_POEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)**FRPOOnBeforeMovePages****Syntax**

```
typedef void (*FRPOOnBeforeMovePages)(
    FS_LPVVOID clientData,
    FR_Document frDoc,
    FS_INT32 nMoveTo,
    FS_WordArray ArrToMove
);
```

**Description**

A callback for page organizing event handler. It is called before the Foxit Reader moves the pdf pages.

**Parameter**


---

clientData	[In] The user-supplied data.
frDoc	[In] The document where the pdf pages are moved.
nMoveTo	[In] The index to move.
ArrToMove	[In] The array to store the pages.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5861

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

## FRPOOnAfterMovePages

**Syntax**

```
typedef void (*FRPOOnAfterMovePages)(  
    FS\_LPVOID clientData,  
    FR\_Document frDoc,  
    FS\_INT32 nMoveTo,  
    FS\_WordArray ArrToMove  
)
```

**Description**

A callback for page organizing event handler. It is called after the Foxit Reader moves the pdf pages.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The document where the pdf pages are moved.
-------	--

---

nMoveTo	[In] The index to move.
---------	-------------------------

---

ArrToMove	[In] The array to store the pages.
-----------	------------------------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5878

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)**FRPOOnRelease****Syntax**

```
typedef void (*FRPOOnRelease)(  
    FS\_LPVOID clientData  
);
```

**Description**

A callback for page organizing event handler. It is called when the Foxit Reader releases the page organizing event handler.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5892

**Group**

[FR\\_POEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRPrefPageOnCreate****Syntax**

```
typedef void (*FRPrefPageOnCreate)(  
    FS\_LPVOID clientData,  
    HWND window  
);
```

**Description**

A callback for Reader preference page. It is called after the preference dialog is created. Client should create a new preference page and register it using [FRAppAddPreferencePage\(\)](#) in this method.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

window	[In] The parent dialog.
--------	-------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 3647

**Group**[FR\\_PreferPageCallbacksRec](#)**FRPrefPageOnDestroy****Syntax**

```
typedef void (*FRPrefPageOnDestroy)(  
    FS_LPVVOID clientData  
)
```

**Description**

A callback for Reader preference page. It is called when user closes the preference dialog. Client should destroy the preference page window and free dynamic memory in this method.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 3658

**Group**[FR\\_PreferPageCallbacksRec](#)**FRPrefPageOnSaveData****Syntax**

```
typedef void (*FRPrefPageOnSaveData)(  
    FS_LPVVOID clientData  
)
```

**Description**

A callback for Reader preference page. It is called when user clicks the *OK* button on the property dialog.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 3667

**Group**[FR\\_PreferPageCallbacksRec](#)**FRPrefPageOnGetTabTitle****Syntax**

```
typedef void (*FRPrefPageOnGetTabTitle)(  
    FS\_LPVOID clientData,  
    FS\_WideString wsTitle  
);
```

**Description**

It is called by Foxit Reader to get the title of the preference page tab.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

wsTitle	[Out] The title of tab.
---------	-------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 3677

**Group**[FR\\_PreferPageCallbacksRec](#)**FRRibbonRecentFileChangItemPinned****Syntax**

```
typedef void (*FRRibbonRecentFileChangItemPinned)(  
    FS\_LPVOID clientData,  
    FS\_LPCSTR lpcsRecentListName,  
    FS\_INT32 nIndex,  
    FS\_LPCWSTR lpwsFilePath,  
    FS\_INT32 nPinned  
);
```

**Description**

---

A callback for Ribbon recent file list event handler. It is called when the item of the recent file list is pinned.

#### Parameter

clientData	[In] The user-supplied data.
lpcsRecentListName	[In] The recent file list name.
nIndex	[In] The item index of the recent file list.
lpwsFilePath	[In] The recent file path.
nPinned	[In] The status of the pin, 0 for not pinned, 1 for pinned.

#### Return

void.

#### Head file reference

fr\_appExpT.h: 5929

#### Group

[FR\\_RibbonRecentFileEventCallbacksRec](#)

#### Since

[SDK LATEEST VERSION > 7.2.2](#)

## FRRibbonRecentFileRemoveItem

#### Syntax

```
typedef void (*FRRibbonRecentFileRemoveItem)(  
    FS\_LPVOID clientData,  
    FS\_LPCSTR lpcsRecentListName,  
    FS\_INT32 nIndex,  
    FS LPCWSTR lpwsFilePath  
);
```

#### Description

A callback for Ribbon recent file list event handler. It is called when the item of the recent file list is removed.

#### Parameter

clientData	[In] The user-supplied data.
------------	------------------------------

---

lpcsRecentListName	[In] The recent file list name.
--------------------	---------------------------------

---

nIndex	[In] The item index of the recent file list.
--------	--

---

lpwsFilePath	[In] The recent file path.
--------------	----------------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 5946

**Group**[FR\\_RibbonRecentFileEventCallbacksRec](#)**Since**[SDK LATEEST VERSION > 7.2.2](#)**FRSecurityOnInit****Syntax**

```
typedef FS_LPVOID (*FRSecurityOnInit)(
    FS_LPVOID clientData,
    FS_LPCWSTR filePath,
    FPD_Object encryptDict,
    FPD_Document doc
);
```

**Description**

It is invoked to initialize the security handler. Plug-ins can initialize the data in this callback. The handler should typically keep the encryption dictionary, and process data in the encryption dictionary.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

filePath	[In] The file path of the PDF document.
----------	---

---

encryptDict	[In] The Encrypt dictionary.
-------------	------------------------------

---

doc	[In] The PDF document.
-----	------------------------

---

**Return**

Non-zero means initializing successfully, otherwise failed. The data returned can be passed through *securityHandler*.

**Head file reference**

fr\_appExpT.h: 2394

**Group**

[FR\\_SecurityCallbacksRec](#)

**FRSecurityIsProcessErrMsg****Syntax**

```
typedef FS_BOOL (*FRSecurityIsProcessErrMsg)(  
    FS_LPVVOID clientData  
)
```

**Description**

Checks whether the plug-in processes the error message.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

whether the plug-in processes the error message.

**Head file reference**

fr\_appExpT.h: 2404

**Group**

[FR\\_SecurityCallbacksRec](#)

**FRSecurityGetPermissions****Syntax**

```
typedef FS_DWORD (*FRSecurityGetPermissions)(  
    FS_LPVVOID clientData,  
    FS_LPVVOID securityData  
)
```

**Description**

Gets permission settings of the document.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

securityData	[In] The data returned from <a href="#">FRSecurityOnInit ()</a> ;
--------------	---

---

**Return**

The permission settings of the document.

**Head file reference**

fr\_appExpT.h: 2415

**Group**

[FR\\_SecurityCallbacksRec](#)

**FRSecurityIsOwner****Syntax**

```
typedef FS_BOOL (*FRSecurityIsOwner)(
    FS_LPVVOID securityData,
    FS_LPVVOID clientData
);
```

**Description**

Checks whether the current user is owner of the document.

**Parameter**


---

securityData	[In] The data returned from <a href="#">FRSecurityOnInit ()</a> ;
--------------	---

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

Whether the current user is owner of the document.

**Head file reference**

fr\_appExpT.h: 2426

**Group**

[FR\\_SecurityCallbacksRec](#)

**FRSecurityGetCryptInfo****Syntax**

```
typedef FS_BOOL (*FRSecurityGetCryptInfo)(
    FS_LPVVOID clientData,
    FS_LPVVOID securityData,
    int* outCipher,
    FS_LPVVOID* outBuffer,
    FS_INT32* outKeylen
);
```

**Description**

Get encryption information including standard algorithm and key.

**Parameter**

clientData	[In] The user-supplied data.
securityData	[In] The data returned from <a href="#">FRSecurityOnInit ()</a> ;
outCipher	[Out] Receives cipher identifier ( <a href="#">FRCIPHER_NONE</a> , <a href="#">FRCIPHER_RC4</a> or <a href="#">FRCIPHER_AES</a> ).
outBuffer	[Out] Receives a pointer to the key buffer.
outKeylen	[Out] Receives number of bytes of the key.

**Return**

[TRUE](#) if successful. [FALSE](#) if no standard key info is provided or failure.

**Head file reference**

fr\_appExpT.h: 2440

**Group**

[FR\\_SecurityCallbacksRec](#)

**FRSecurityIsMetadataEncrypted****Syntax**

```
typedef FS_BOOL (*FRSecurityIsMetadataEncrypted)(  
    FS\_LPVVOID securityData,  
    FS\_LPVVOID clientData  
)
```

**Description**

Checks if document metadata needs to be encrypted.

**Parameter**

securityData	[In] The data returned from <a href="#">FRSecurityOnInit ()</a> ;
clientData	[In] The user-supplied data.

**Return**

Whether document metadata needs to be encrypted or not.

**Head file reference**

fr\_appExpT.h: 2452

**Group**[FR\\_SecurityCallbacksRec](#)**FRSecurityFinishHandler****Syntax**

```
typedef void (*FRSecurityFinishHandler)(  
    FS\_LPVVOID clientData,  
    FS\_LPVVOID securityData  
);
```

**Description**

Release the security handler.

**Parameter**

clientData	[In] The user-supplied data.
securityData	[In] The data returned from <a href="#">FRSecurityOnInit ()</a> ;

**Return**

void

**Head file reference**

fr\_appExpT.h: 2463

**Group**[FR\\_SecurityCallbacksRec](#)**FRSecurityCreateCryptoHandler****Syntax**

```
typedef FS_LPVOID (*FRSecurityCreateCryptoHandler)(  
    FS\_LPVVOID clientData,  
    FS\_LPVVOID securityData,  
    FS\_INT32* outType  
);
```

**Description**Create a crypto handler that can do the real encryption/decryption work. After creation, the caller should call [FREnryptCreateHandler \(\)](#) to initialize the crypto handler.**Parameter**

---

clientData	[In] The user-supplied data.
securityData	[In] The data returned from <a href="#">FRSecurityOnInit()</a> ;
outType	[Out] It identifies the type of crypto handler.

---

**Return**

If *outType* is 0, this callback returns the pointer to structure containing the crypto handler callbacks [FR\\_CryptoCallbacks](#). If *outType* is 1, this callback returns the pointer to internal crypto handler.

**Head file reference**

fr\_appExpT.h: 2477

**Group**

[FR\\_SecurityCallbacksRec](#)

**FRSecurityMethodGetName****Syntax**

```
typedef FS_LPWSTR (*FRSecurityMethodGetName)(
    FS_LPVVOID clientData
);
```

**Description**

It is called by Foxit Reader to get the name of the security method.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

The name of the security method.

**Head file reference**

fr\_appExpT.h: 4946

**Group**

[FR\\_SecurityMethodCallbacksRec](#)

**FRSecurityMethodGetTitle****Syntax**

```
typedef FS_LPWSTR (*FRSecurityMethodGetTitle)(
    FS_LPVVOID clientData
);
```

**Description**

It is called by Foxit Reader to get the title of the security method.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

The title of the security method.

**Head file reference**

fr\_appExpT.h: 4956

**Group**

[FR\\_SecurityMethodCallbacksRec](#)

**FRSecurityMethodIsMyMethod****Syntax**

```
typedef FS_BOOL (*FRSecurityMethodIsMyMethod)(  
    FS_LPVVOID clientData,  
    FR_Document doc  
>;
```

**Description**

It is called by Foxit Reader to ask the plug-in whether the security applied for the document can be handled.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] The document.
-----	--------------------

---

**Return**

[TRUE](#) if the security applied for the document can be handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 4967

**Group**

[FR\\_SecurityMethodCallbacksRec](#)

**FRSecurityMethodCheckModuleLicense****Syntax**

```
typedef FS_BOOL (*FRSecurityMethodCheckModuleLicense)(  
    FS_LPVVOID clientData  
);
```

**Description**

It is called by Foxit Reader to check whether the plug-in has the license to modify the security. In general, it is a internal interface.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

[TRUE](#) if the plug-in has the license to modify the security, otherwise not.

**Head file reference**

fr\_appExpT.h: 4978

**Group**

[FR\\_SecurityMethodCallbacksRec](#)

**FRSecurityMethodCanBeModified****Syntax**

```
typedef FS_BOOL (*FRSecurityMethodCanBeModified)(  
    FS_LPVVOID clientData  
);
```

**Description**

It is called by Foxit Reader to check whether the plug-in has the right to modify the security.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

[TRUE](#) if the plug-in has the right to modify the security, otherwise not.

**Head file reference**

fr\_appExpT.h: 4989

**Group**

[FR\\_SecurityMethodCallbacksRec](#)

**FRSecurityMethodDoSetting**

**Syntax**

```
typedef void (*FRSecurityMethodDoSetting)(  
    FS_LPVOID clientData,  
    HWND hWnd,  
    FS_BOOL* bSuc  
) ;
```

**Description**

It is called by Foxit Reader when the user clicks the setting button in the security page of document property dialog. You can implement your own setting process.

**Parameter**


---

clientData	[In] The user-supplied data.
hWnd	[In] The parent window used to create the child window.
bSuc	[Out] It receives the result whether the setting is successful.

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 5002

**Group**

[FR\\_SecurityMethodCallbacksRec](#)

**FRSecurityMethodRemoveSecurityInfo****Syntax**

```
typedef FS_BOOL (*FRSecurityMethodRemoveSecurityInfo)(  
    FS_LPVOID clientData  
) ;
```

**Description**

It is called by Foxit Reader when the security method is to be removed. Then you can implement your own removing process.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

[TRUE](#) if you remove the security method correctly, otherwise not.

**Head file reference**

---

fr\_appExpT.h: 5012

**Group**[FR\\_SecurityMethodCallbacksRec](#)**FRSecurityMethodCreatePermSubDlg****Syntax**

```
typedef HWND (*FRSecurityMethodCreatePermSubDlg)(
    FS\_LPVVOID clientData,
    HWND hParent
);
```

**Description**

It is called by Foxit Reader when the security page of document property dialog is to be shown. If the security method is yours, you have to create a child window to show the information of the security method.

**Parameter**


---

clientData	[In] The user-supplied data.
hParent	[In] The parent window used to create the child window.

---

**Return**

The handle of the child window used to show the information of the security method.

**Head file reference**

fr\_appExpT.h: 5024

**Group**[FR\\_SecurityMethodCallbacksRec](#)**FRSecurityMethodDestroyPermSubDlg****Syntax**

```
typedef void (*FRSecurityMethodDestroyPermSubDlg)(
    FS\_LPVVOID clientData,
    HWND hWnd
);
```

**Description**

It is called by Foxit Reader when the security page of document property dialog is to be destroyed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

hWnd	[In] The window handle
------	------------------------

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 5035

**Group**[FR\\_SecurityMethodCallbacksRec](#)**FRSelectionGetType****Syntax**

```
typedef FS_LPSTR (*FRSelectionGetType)(
    FS_LPVVOID clientData
);
```

**Description**

It returns the selection type (for example, 'Text' or 'Bookmark'). This information is used so that the Foxit Reader knows which selection handler to call.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

The selection type.

**Head file reference**

fr\_appExpT.h: 3737

**Group**[FR\\_SelectionCallbacksRec](#)**FRSelectionCanSelectAll****Syntax**

```
typedef FS_BOOL (*FRSelectionCanSelectAll)(
    FS_LPVVOID clientData,
    FR_Document document,
    void* curSelectData
);
```

**Description**

It is used to determine whether the current selection type can perform a select all operation. This controls whether the Select All menu item is enabled.

### Parameter

clientData	[In] The user-supplied data.
document	[In] The document containing the current selection.
curSelectData	[In] The current selection's data.

### Return

[TRUE](#) if select all can be performed on the current selection type, [FALSE](#) otherwise.

#### Head file reference

fr\_appExpT.h: 3750

### Group

[FR\\_SelectionCallbacksRec](#)

## FRSelectionDoSelectAll

### Syntax

```
typedef void (*FRSelectionDoSelectAll)(  
    FS\_LPVVOID clientData,  
    FR\_Document document,  
    void* curSelectData  
);
```

### Description

It is called to perform the select all operation.

### Parameter

clientData	[In] The user-supplied data.
document	[In] The document in which the <i>Select All</i> is performed.
curSelectData	[In] The current selection data in doc.

### Return

#### Head file reference

fr\_appExpT.h: 3762

**Group**[FR\\_SelectionCallbacksRec](#)**FRSelectionCanDelete****Syntax**

```
typedef FS_BOOL (*FRSelectionCanDelete)(  
    FS_LPVVOID clientData,  
    FR_Document document,  
    void* curSelectData  
)
```

**Description**

It is used to determine whether the current selection can be deleted. This controls, for example, whether the *Delete* menu item is enabled.

**Parameter**

clientData	[In] The user-supplied data.
document	[In] The document containing the current selection.
curSelectData	[In] The current selection data.

**Return**

[TRUE](#) if the current selection can be deleted, [FALSE](#) otherwise.

**Head file reference**

fr\_appExpT.h: 3775

**Group**[FR\\_SelectionCallbacksRec](#)**FRSelectionToDelete****Syntax**

```
typedef void (*FRSelectionToDelete)(  
    FS_LPVVOID clientData,  
    FR_Document document,  
    void* curSelectData  
)
```

**Description**

It deletes the current selection.

**Parameter**

---

clientData	[In] The user-supplied data.
document	[In] Document whose selection is deleted.
curSelectData	[In] The current selection in doc.

---

**Return****Head file reference**

fr\_appExpT.h: 3787

**Group**[FR\\_SelectionCallbacksRec](#)**FRSelectionCanCopy****Syntax**

```
typedef FS_BOOL (*FRSelectionCanCopy)(  
    FS\_LPVVOID clientData,  
    FR\_Document document,  
    void* curSelectData  
)
```

**Description**

It is used to determine whether the current selection can be copied. This controls, for example, whether the *Copy* menu item is enabled.

**Parameter**


---

clientData	[In] The user-supplied data.
document	[In] The document containing the selection.
curSelectData	[In] The current selection data.

---

**Return**

[TRUE](#) if the current selection can be copied, [FALSE](#) otherwise.

**Head file reference**

fr\_appExpT.h: 3800

**Group**[FR\\_SelectionCallbacksRec](#)**FRSelectionDoCopy**

**Syntax**

```
typedef void (*FRSelectionDoCopy)(  
    FS\_LPVVOID clientData,  
    FR\_Document document,  
    void* curSelectData  
);
```

**Description**

It copies the selected item to the clipboard. The Foxit Reader viewer will have already cleared the clipboard and placed some private data onto it, in order to identify the selectionhandler that put data on the clipboard. Because of this, a plug-in must not clear the clipboard, and should only add its private data. In addition, if the current selection can be reasonably represented as text, plug-ins are strongly encouraged to place a text representation of the selection onto the clipboard, in addition to their own private format.

**Parameter**

clientData	[In] The user-supplied data.
document	[In] The document whose selection is copied.
curSelectData	[In] The current selection data in doc.

**Return****Head file reference**

fr\_appExpT.h: 3819

**Group**[FR\\_SelectionCallbacksRec](#)**FRSelectionCanCut****Syntax**

```
typedef FS_BOOL (*FRSelectionCanCut)(  
    FS\_LPVVOID clientData,  
    FR\_Document document,  
    void* curSelectData  
);
```

**Description**

It is used to determine whether the current selection can be cut. This controls, for example, whether the *Cut* menu item is enabled.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

---

document	[In] The document containing the current selection.
----------	---

---

curSelectData	[In] The current selection data.
---------------	----------------------------------

---

**Return**

[TRUE](#) if the current selection can be cut, [FALSE](#) otherwise.

**Head file reference**

fr\_appExpT.h: 3832

**Group**

[FR\\_SelectionCallbacksRec](#)

**FRSelectionDoCut****Syntax**

```
typedef void (*FRSelectionDoCut)(
    FS\_LPVOID clientData,
    FR\_Document document,
    void* curSelectData
);
```

**Description**

It cuts the current selection. See the discussion under [FRSelectionDoCopy\(\)](#) for information on how the selection handler must use the clipboard.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

document	[In] Document whose selection is cut.
----------	---------------------------------------

---

curSelectData	[In] The current selection data in doc.
---------------	---

---

**Return****Head file reference**

fr\_appExpT.h: 3845

**Group**

[FR\\_SelectionCallbacksRec](#)

**FRSelectionCanPaste****Syntax**

```
typedef FS_BOOL (*FRSelectionCanPaste)(
```

```
FS_LPVVOID clientData,  
FR_Document document  
);
```

**Description**

It is used to determine whether the current selection can be pasted. This controls, for example, whether the *Paste* menu item is enabled.

**Parameter**

clientData	[In] The user-supplied data.
document	[In] The document into which the selection is pasted.

**Return**

[TRUE](#) if the data currently on the clipboard can be pasted, [FALSE](#) otherwise.

**Head file reference**

fr\_appExpT.h: 3857

**Group**

[FR\\_SelectionCallbacksRec](#)

**FRSelectionDoPaste****Syntax**

```
typedef void (*FRSelectionDoPaste)(  
    FS_LPVVOID clientData,  
    FR_Document document  
,
```

**Description**

It pastes the current selection from the clipboard.

**Parameter**

clientData	[In] The user-supplied data.
document	[In] The document into whose selection the clipboard is pasted.

**Return****Head file reference**

fr\_appExpT.h: 3868

**Group**

[FR\\_SelectionCallbacksRec](#)**FRSelectionLosingSelection****Syntax**

```
typedef void (*FRSelectionLosingSelection)(
    FS_LPVVOID clientData,
    FR_Document document,
    void* curSelectData
);
```

**Description**

This method is called by [FRSelectionClearSelection](#) (among others), to let the selection handler responsible for the old selection do whatever cleanup it needs.

**Parameter**

clientData	[In] The user-supplied data.
document	[In] The document whose selection is cleared.
curSelectData	[In] The current selection data in doc.

**Return****Head file reference**

fr\_appExpT.h: 3881

**Group**[FR\\_SelectionCallbacksRec](#)**FRSelectionGettingSeletion****Syntax**

```
typedef void (*FRSelectionGettingSeletion)(
    FS_LPVVOID clientData,
    FR_Document document,
    void* curSelectData
);
```

**Description**

It is called when the selection is set (for example, via FRSelectionSetCurSelection()).

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

---

document	[In] The document containing the selection.
----------	---

---

curSelectData	[In] The selection data being added.
---------------	--------------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 3893

**Group**[FR\\_SelectionCallbacksRec](#)**FRSelectionRemovedFromSelection****Syntax**

```
typedef void* (*FRSelectionRemovedFromSelection)(
    FS\_LPVVOID clientData,
    FR\_Document document,
    void* curSelectData,
    void* remData
);
```

**Description**

A callback that de-highlights the old item given in remData, and returns a new *curSelectData* or [NULL](#) if failure occurred.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

document	[In] The document in which an item is removed from the selection.
----------	---

---

curSelectData	[In] The current selection data.
---------------	----------------------------------

---

remData	[In] The item to remove from the selection. The content and format of selData differsfor each selection handler, and are decided by the selection handler's implementors.
---------	---

---

**Return**

The new selection data after the specified item has been removed.

**Head file reference**

fr\_appExpT.h: 3908

**Group**

[FR\\_SelectionCallbacksRec](#)**FRSelectionAddedToSelection****Syntax**

```
typedef void* (*FRSelectionAddedToSelection)(  
    FS_LPVVOID clientData,  
    FR_Document document,  
    void* curSelectData,  
    void* addData  
);
```

**Description**

A callback that adds the specified item to the selection, highlights it, and returns the new selection containing the newly-added item.

**Parameter**


---

clientData	[In] The user-supplied data.
document	[In] The document containing the data to add to the selection.
curSelectData	[In] Data representing the current selection. Its format is specific to the selection handler.
addData	[In] The item to add to the selection.

---

**Return**

New selection data containing all current selections (that is, the previous selection plus the newly-added selection), or [NULL](#) if failure occurred. If the selection handler allows only a single item to be selected at a time, clear the previous selection, highlight the selection specified by *addData* (if *highlight* is true), and simply return *addData*.

**Head file reference**

fr\_appExpT.h: 3925

**Group**[FR\\_SelectionCallbacksRec](#)**FRSelectionShowSelection****Syntax**

```
typedef void (*FRSelectionShowSelection)(  
    FS_LPVVOID clientData,  
    FR_Document document,  
    void* curSelectData  
);
```

**Description**

It changes the view (for example, by scrolling the current page or moving to the appropriate page) so that the current selection is visible.

**Parameter**

clientData	[In] The user-supplied data.
document	[In] The document whose selection is displayed.
curSelectData	[In] The current selection data in doc.

**Return****Head file reference**

fr\_appExpT.h: 3938

**Group**

[FR\\_SelectionCallbacksRec](#)

**FRSelectionKeyDown****Syntax**

```
typedef FS_BOOL (*FRSelectionKeyDown)(  
    FS_LPVVOID clientData,  
    FR_Document document,  
    void* curSelectData,  
    unsigned int nKeyCode,  
    unsigned int nFlags  
)
```

**Description**

It handles a key press. It is needed only if the selection handler processes key presses.

**Parameter**

clientData	[In] The user-supplied data.
document	[In] The document in which the click occurred.
curSelectData	[In] The current selection data for doc.
nKeyCode	[In] The key that was pressed.

---

nFlags	[In] Modifier keys that were pressed with key. It must be an OR of the Modifier Keys values.
--------	--

---

**Return**

[TRUE](#) if it the keypress was handled, [FALSE](#) if it was not and therefore needs to be passed to the nextprocedure in the key handling chain.

**Head file reference**

fr\_appExpT.h: 3953

**Group**

[FR\\_SelectionCallbacksRec](#)

**FRSelectionKeyUp****Syntax**

```
typedef FS_BOOL (*FRSelectionKeyUp)(  
    FS\_LPVVOID clientData,  
    FR\_Document document,  
    void* curSelectData,  
    unsigned int nKeyCode,  
    unsigned int nFlags  
>;
```

**Description**

It handles a key press. It is needed only if the selection handler processes key presses.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

document	[In] The document in which the click occurred.
----------	--

---

curSelectData	[In] The current selection data for doc.
---------------	--

---

nKeyCode	[In] The key that was pressed.
----------	--------------------------------

---

nFlags	[In] Modifier keys that were pressed with key. It must be an OR of the Modifier Keys values.
--------	--

---

**Return**

[TRUE](#) if it the keypress was handled, [FALSE](#) if it was not and therefore needs to be passed to the nextprocedure in the key handling chain.

**Head file reference**

fr\_appExpT.h: 3968

**Group**[FR\\_SelectionCallbacksRec](#)**FRSelectionKeyChar****Syntax**

```
typedef FS_BOOL (*FRSelectionKeyChar)(
    FS\_LPVOID clientData,
    FR\_Document document,
    void* curSelectData,
    unsigned int nChar,
    unsigned int nFlags
);
```

**Description**

It handles a key press. It is needed only if the selection handler processes key presses.

**Parameter**

clientData	[In] The user-supplied data.
document	[In] The document in which the click occurred.
curSelectData	[In] The current selection data for doc.
nChar	[In] Contains the character code value of the key.

**nFlags**

[In] Contains the scan code, key-transition code, previous key state, and context code.

**Return**

[TRUE](#) if it the keypress was handled, [FALSE](#) if it was not and therefore needs to be passed to the nextprocedure in the key handling chain.

**Head file reference**

fr\_appExpT.h: 3983

**Group**[FR\\_SelectionCallbacksRec](#)**FRSelectionMouseWheel****Syntax**

```
typedef FS_BOOL (*FRSelectionMouseWheel)(
    FS\_LPVOID clientData,
```

---

```
FR_PageView pageView,
void* curSelectData,
unsigned int nFlags,
const FS_DevicePoint point
);
```

**Description**

It handles a mouse wheeling. It is needed only if the selection handler processes mouse wheeling.

**Parameter**

clientData	[In] The user-supplied data.
pageView	[In] The page view in which the mouse wheeling occurred.
curSelectData	[In] The current selection data for doc.
nFlags	[In] Indicates whether various virtual keys are down.
point	[In] Specifies the x- and y-coordinate of the cursor.These coordinates are always relative to the upper-left corner of the window.

**Return**

[TRUE](#) if the mouse wheeling was handled, otherwise not.

**Head file reference**

fr\_appExpT.h: 3998

**Group**

[FR\\_SelectionCallbacksRec](#)

**FRSelectionCanDeselectAll****Syntax**

```
typedef FS_BOOL (*FRSelectionCanDeselectAll)(
    FS_LPVVOID clientData,
    FR_Document document,
    void* curSelectData
);
```

**Description**

It is used to determine whether all the current selections can be deselected.This controls, for example, whether the *Deselect All* menu item is enabled.

**Parameter**


---

clientData	[In] The user-supplied data.
document	[In] The document containing the current selection.
curSelectData	[In] The current selection data.

---

**Return**

[TRUE](#) if all the data currently can be deselected, [FALSE](#) otherwise.

**Head file reference**

fr\_appExpT.h: 4011

**Group**

[FR\\_SelectionCallbacksRec](#)

**FRSelectionDoDeselectAll****Syntax**

```
typedef void (*FRSelectionDoDeselectAll)(  
    FS\_LPVOID clientData,  
    FR\_Document document,  
    void* curSelectData  
>;
```

**Description**

It deselects all the items selected.

**Parameter**


---

clientData	[In] The user-supplied data.
document	[In] The document containing the current selection.
curSelectData	[In] The current selection data.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 4023

**Group**

[FR\\_SelectionCallbacksRec](#)

## FRSignatureHandlerGetName

### Syntax

```
typedef char* (*FRSignatureHandlerGetName)(  
    FS\_LPVOID clientData  
);
```

### Description

A callback for signature handler. It is called to provide the name of the signature handler.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

### Return

The name of the signature handler.

### Head file reference

fr\_docExpT.h: 524

### Group

[FR\\_SignatureHandlerCallbacksRec](#)

### Since

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

## FRSignatureHandlerSignData

### Syntax

```
typedef FR_SG_HANDLE (*FRSignatureHandlerSignData)(  
    FS\_LPVOID clientData,  
    const unsigned char* pData2BSigned,  
    unsigned long ulData2BSignedLen,  
    unsigned char** pPSignedData,  
    unsigned long* pulSignedDataLen  
);
```

### Description

A callback for signature handler. It is called to by Foxit Reader to pass the data to be signed to the plug-in.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

pData2BSigned	[In] The data to be signed.
---------------	-----------------------------

---

---

ulData2BSignedLen	[In] The length of the data to be signed.
-------------------	---

---

pPSignedData	[Out] Pass the signed data back to the framework.
--------------	---

---

pulSignedDataLen	[Out] Pass the length of the signed data back to the framework.
------------------	---

---

**Return**

The result of signing the data.

**Head file reference**

fr\_docExpT.h: 542

**Group**

[FR\\_SignatureHandlerCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 7.2.2](#)

**FRSignatureHandlerTimeStampDate****Syntax**

```
typedef FR_SG_HANDLE (*FRSignatureHandlerTimeStampDate)(
    FS\_LPVVOID clientData,
    FR\_SignatureTimestamp* pSgTimeStampDate
);
```

**Description**

A callback for signature handler. It is called to by Foxit Reader to receive the timestamp date from plug-in.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

pSgTimeStampDate	[Out] Receives the timestamp date from plug-in.
------------------	---

---

**Return**

The result of getting the timestamp date.

**Head file reference**

fr\_docExpT.h: 557

**Group**

[FR\\_SignatureHandlerCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

## FRSignatureHandlerVerifyData

**Syntax**

```
typedef FR_SG_HANDLE (*FRSignatureHandlerVerifyData)(  
    FS_LPVOID clientData,  
    const unsigned char* pSignedData,  
    unsigned long ulSignedDataLen,  
    const unsigned char* pData2BSigned,  
    unsigned long ulDataT2BSignedLen,  
    FS_DWORD* outVerifyState,  
    void** hCertContext  
) ;
```

**Description**

A callback for signature handler. It is called to by Foxit Reader to pass the data to the plug-in. Then the plug-in can takes over the process for verifying the signed data.

**Parameter**


---

clientData	[In] The user-supplied data.
pSignedData	[In] The signed data.
ulSignedDataLen	[In] The length of the signed data.
pData2BSigned	[In] The source data that is signed.
ulDataT2BSignedLen	[In] The length of the source data that is signed.
outVerifyState	[Out] Pass the verifying state back to framework. See <a href="#">FRSIGVarifySignatureStates</a> .
hCertContext	[Out] The certificate context. It will be passed to <a href="#">FRSignatureHandlerShowStateUI</a> ;

---

**Return**

The result of verifying the data. If the plug-in returns [FR SIG HANDLE NONE](#) , the Foxit Reader will verify the data by default method.

**Head file reference**

fr\_docExpT.h: 577

**Group**

[FR\\_SignatureHandlerCallbacksRec](#)

**Since**  
[SDK LATEEST VERSION > 7.2.2](#)

## FRSignatureHandlerShowStateUI

### Syntax

```
typedef FR_SG_HANDLE (*FRSignatureHandlerShowStateUI)(
    FS\_LPVVOID clientData,
    const FS\_DWORD nVerifyState,
    void* pWnd,
    void* hCertContext,
    FR\_SignatureDictInfo* pSignDictInfo
);
```

### Description

A callback for signature handler. It is called to by Foxit Reader to show the state UI. The plug-in can create its own UI.

### Parameter

clientData	[In] The user-supplied data.
nVerifyState	[In] The verifying state.
pWnd	[In] The parent window of the state UI. It represents the <i>MFC CWnd</i> *.
hCertContext	[In] The certificate context from <a href="#">FRSignatureHandlerVerifyData</a> .
pSignDictInfo	[In] The signature dictionary info.

### Return

The result of showing state UI.

**Head file reference**  
fr\_docExpT.h: 596

**Group**  
[FR\\_SignatureHandlerCallbacksRec](#)

**Since**  
[SDK LATEEST VERSION > 7.2.2](#)

## FRSignatureHandlerCanClear

**Syntax**

```
typedef FR_SG_HANDLE (*FRSignatureHandlerCanClear)(
    FS\_LPVOID clientData,
    void* hCertContext,
    FS\_BOOL* bCanClear
);
```

**Description**

A callback for signature handler. It is called to by Foxit Reader to check whether the signature can be cleared or not.

**Parameter**

clientData	[In] The user-supplied data.
hCertContext	[In] The certificate context from <a href="#">FRSignatureHandlerVerifyData</a> .
bCanClear	[Out] Whether the signature can be cleared or not.

**Return**

The result of this checking.

**Head file reference**

fr\_docExpT.h: 612

**Group**

[FR\\_SignatureHandlerCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 7.2.2](#)

**FRSignatureHandlerShowSignProperties****Syntax**

```
typedef FR_SG_HANDLE (*FRSignatureHandlerShowSignProperties)(
    FS\_LPVOID clientData,
    const unsigned char* pSignedData,
    unsigned long ulSignedDataLen,
    const unsigned char* pData2BSigned,
    unsigned long ulDataT2BSignedLen,
    void* pWnd,
    FR\_SignatureDictInfo* pSignDictInfo,
    void* hCertContext
);
```

**Description**

A callback for signature handler. It is called to by Foxit Reader to show the signing properties. The plug-in can create its own UI.

**Parameter**


---

clientData	[In] The user-supplied data.
pSignedData	[In] The signed data.
ulSignedDataLen	[In] The length of the signed data.
pData2BSigned	[In] The source data that is signed.
ulDataT2BSignedLen	[In] The length of the source data that is signed.
pWnd	[In] The parent window of the property UI. It represents the <i>MFC CWnd*</i> .
pSignDictInfo	[In] The signature dictionary info.
hCertContext	[In] The certificate context from <a href="#">FRSignatureHandlerVerifyData</a> .

---

**Return**

The result of showing the signing property UI.

**Head file reference**

fr\_docExpT.h: 633

**Group**

[FR\\_SignatureHandlerCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**RSignatureHandlerReleaseSignData****Syntax**

```
typedef void (*RSignatureHandlerReleaseSignData)(
    unsigned char pPSignedData,
    unsigned long pulSignedDataLen
);
```

**Description**

A callback for signature handler. It is called to by Foxit Reader to Release the data to be signed.

**Parameter**

---

pPSignedData	[In] The data to be Released.
--------------	-------------------------------

---

pulSignedDataLen	[In] The length of the data to be Released.
------------------	---

---

**Return****Head file reference**

fr\_docExpT.h: 649

**Group**[FR\\_SignatureHandlerCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 9.1](#)**FRStatusBarWndExOnGetTooltip****Syntax**

```
typedef void (*FRStatusBarWndExOnGetTooltip)(  
    FS\_LPVOID clientData,  
    FS\_WideString outTooltip  
>);
```

**Description**

A callback for adding a window to the status bar. The callback is invoked by Foxit Reader to receive the tooltip of the status bar window.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

outTooltip	[Out] It receives the tooltip.
------------	--------------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1471

**Group**[FR\\_StatusBarWndExCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRStatusBarWndExOnCreateWnd****Syntax**

```
typedef HWND (*FRStatusBarWndExOnCreateWnd)(  
    FS_LPVVOID clientData,  
    void* pParent  
);
```

**Description**

A callback for adding a window to the status bar. The callback is invoked by Foxit Reader to notify the plug-in to create the window.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

pParent	[In] The parent window. It represents the <i>MFC CWnd</i> .
---------	---

---

**Return**

The status bar window created.

**Head file reference**

fr\_barExpT.h: 1486

**Group**

[FR\\_StatusBarWndExCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRStatusBarWndExGetSize****Syntax**

```
typedef void (*FRStatusBarWndExGetSize)(  
    FS_LPVVOID clientData,  
    FS_INT32* cx,  
    FS_INT32* cy  
);
```

**Description**

A callback for adding a window to the status bar. The callback is invoked by Foxit Reader to receive the size of the status bar window.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

<code>cx</code>	[Out] It receives the horizon size.
-----------------	-------------------------------------

---

<code>cy</code>	[Out] It receives the vertical size.
-----------------	--------------------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1502

**Group**[FR\\_StatusBarWndExCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRStatusBarWndExOnSize****Syntax**

```
typedef void (*FRStatusBarWndExOnSize)(
    FS\_LPVOID clientData,
    FS\_INT32 cx,
    FS\_INT32 cy
);
```

**Description**

A callback for adding a window to the status bar. The callback is invoked by Foxit Reader when the size is changed.

**Parameter**


---

<code>clientData</code>	[In] The user-supplied data.
-------------------------	------------------------------

---

<code>cx</code>	[In] It indicates the horizon size.
-----------------	-------------------------------------

---

<code>cy</code>	[In] It indicates the vertical size.
-----------------	--------------------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1518

**Group**[FR\\_StatusBarWndExCallbacksRec](#)

**Since**[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

## FRShowSubscribeRibbonUI

**Syntax**

```
typedef FS_BOOL (*FRShowSubscribeRibbonUI)(  
    FS_LVOID clientData,  
    FRSubscriptionFlowName subWorkflowName  
)
```

**Description**

A callback for subscription provider. It is called to notify the plug-in to check that whether to show the subscription UI or not.

**Parameter**

clientData	[In] The user-supplied data.
subWorkflowName	[In] The specified work flow name.

**Return**

TRUE for showing the subscription UI, otherwise not

**Head file reference**

fr\_appExpT.h: 5988

**Group**[FR\\_SubscriptionProviderCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

## FRStartSubscribeWorkflow

**Syntax**

```
typedef FS_BOOL (*FRStartSubscribeWorkflow)(  
    FS_LVOID clientData,  
    FRSubscriptionFlowName subWorkflowName,  
    FS_WideString outReturnValue  
)
```

**Description**

A callback for subscription provider. It is called when a subscription work flow starts.

**Parameter**

---

clientData	[In] The user-supplied data.
subWorkflowName	[In] The specified work flow name.
outReturnValue	[Out] It receives the result.

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_appExpT.h: 6004

**Group**

[FR\\_SubscriptionProviderCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRIsLicenseRevoked****Syntax**

```
typedef FS_BOOL (*FRIsLicenseRevoked)(  
    FS\_LPVOID clientData  
)
```

**Description**

A callback for subscription provider. It is called to notify the plug-in to check that whether the license is revoked or not.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

TRUE means the license is revoked, otherwise not.

**Head file reference**

fr\_appExpT.h: 6018

**Group**

[FR\\_SubscriptionProviderCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRShowSubscribeFlash**

**Syntax**

```
typedef FS_BOOL (*FRShowSubscribeFlash)(  
    FS_LPVVOID clientData  
>);
```

**Description**

A callback for subscription provider. It is called to notify the plug-in to show the subscription flash.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

TRUE when succeed to show the subscription flash, otherwise not.

**Head file reference**

fr\_appExpT.h: 6032

**Group**

[FR\\_SubscriptionProviderCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**OnUndo****Syntax**

```
typedef void (*OnUndo)(  
    FS_LPVVOID clientData  
>);
```

**Description**

It is called to implement the undo operation.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 4414

**Group**

[FR\\_UndoRedoCallbacksRec](#)

**Since**

## [SDK\\_LATEEST\\_VERSION > 2.0](#)

### OnRedo

#### Syntax

```
typedef void (*OnRedo)(  
    FS_LPVOID clientData  
)
```

#### Description

It is called to implement the redo operation.

#### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

#### Return

void

#### Head file reference

fr\_appExpT.h: 4425

#### Group

[FR\\_UndoRedoCallbacksRec](#)

#### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

### OnRelease

#### Syntax

```
typedef void (*OnRelease)(  
    FS_LPVOID clientData  
)
```

#### Description

It is called when the undo-redo item is to be released by Foxit Reader.

#### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

#### Return

void

#### Head file reference

fr\_appExpT.h: 4436

**Group**[FR\\_UndoRedoCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRTTaskPaneViewGetName****Syntax**

```
typedef FS_LPSTR (*FRTTaskPaneViewGetName)(  
    FS_LPVVOID clientData  
)
```

**Description**

It is called by Foxit Reader to receive the name of task pane view you want to create.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

The name of task pane view.

**Head file reference**

fr\_appExpT.h: 4502

**Group**[FR\\_WinMSGCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FRTTaskPaneViewGetTitle****Syntax**

```
typedef FS_LPWSTR (*FRTTaskPaneViewGetTitle)(  
    FS_LPVVOID clientData  
)
```

**Description**

It is called by Foxit Reader to receive the title of task pane view you want to create.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

The title of task pane view.

**Head file reference**

fr\_appExpT.h: 4513

**Group**

[FR\\_WinMSGCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRTTaskPaneViewGetDropMenuItemText

**Syntax**

```
typedef FS_LPWSTR (*FRTTaskPaneViewGetDropMenuItemText)(  
    FS\_LPVOID clientData  
)
```

**Description**

It is called by Foxit Reader to receive the text of drop-down menu at the right-up corner.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

The text of drop-down menu at the right-up corner.

**Head file reference**

fr\_appExpT.h: 4524

**Group**

[FR\\_WinMSGCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRTTaskPaneViewGetDropMenuItemTip

**Syntax**

```
typedef FS_LPWSTR (*FRTTaskPaneViewGetDropMenuItemTip)(  
    FS\_LPVOID clientData  
)
```

**Description**

It is called by Foxit Reader to receive the tooltip of drop-down menu at the right-up corner.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

The tooltip of drop-down menu at the right-up corner.

**Head file reference**

fr\_appExpT.h: 4535

**Group**

[FR\\_WinMSGCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRTTaskPaneViewCreateHwnd****Syntax**

```
typedef HWND (*FRTTaskPaneViewCreateHwnd)(  
    FS\_LPVOID clientData,  
    HWND hOwner  
) ;
```

**Description**

It is called by Foxit Reader to receive the window handle of task pane view. Once the document is opened in browser, this callback will be called to create a window of task pane view.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

hOwner	[In] The parent window used to create the task pane view.
--------	---

---

**Return**

The window handle of task pane view.

**Head file reference**

fr\_appExpT.h: 4548

**Group**

[FR\\_WinMSGCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 1.0](#)

## FRTTaskPaneViewDestroyHwnd

### Syntax

```
typedef void (*FRTTaskPaneViewDestroyHwnd)(  
    FS_LPVOID clientData,  
    HWND hOwner  
);
```

### Description

It is called by Foxit Reader to destroy the window handle of task pane view. Once the document is closed in browser, this callback will be called to destroy the window of task pane view.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

hOwner	[In] The parent window used to create the task pane view.
--------	---

---

### Return

void.

### Head file reference

fr\_appExpT.h: 4561

### Group

[FR\\_WinMSGCallbacksRec](#)

### Since

[SDK LATEEST VERSION > 1.0](#)

## FRTTaskPaneViewGetBigIcon

### Syntax

```
typedef FS_DIBitmap (*FRTTaskPaneViewGetBigIcon)(  
    FS_LPVOID clientData  
);
```

### Description

This interface is reserved.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---



**Return**

This interface is reserved. You can return NULL.

**Head file reference**

fr\_appExpT.h: 4572

**Group**

[FR\\_WinMSGCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRTTaskPaneViewGetSmallIcon

**Syntax**

```
typedef FS_DIBitmap (*FRTTaskPaneViewGetSmallIcon)(  
    FS\_LPVOID clientData  
)
```

**Description**

This interface is reserved.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

This interface is reserved. You can return NULL.

**Head file reference**

fr\_appExpT.h: 4583

**Group**

[FR\\_WinMSGCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRTTaskPaneViewOnActive

**Syntax**

```
typedef void (*FRTTaskPaneViewOnActive)(  
    FS\_LPVOID clientData,  
    HWND hOwner  
)
```

**Description**

It is called when the task pane view is to be activated.

**Parameter**

clientData	[In] The user-supplied data.
hOwner	[In] The parent window of the task pane view.

**Return**

void

**Head file reference**

fr\_appExpT.h: 4595

**Group**

[FR\\_WinMSGCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRTTaskPaneViewOnDestroy****Syntax**

```
typedef void (*FRTTaskPaneViewOnDestroy)(  
    FS_LPVOID clientData  
)
```

**Description**

It is called when the task pane view is to be destroyed.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

**Return**

void

**Head file reference**

fr\_appExpT.h: 4606

**Group**

[FR\\_WinMSGCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRTTaskPaneViewOnShowTaskPane

### Syntax

```
typedef void (*FRTTaskPaneViewOnShowTaskPane)(  
    FS_LPVVOID clientData,  
    HWND hOwner,  
    FS_BOOL bShow  
>);
```

### Description

It is called when the task pane view is to be shown or to be hidden.

### Parameter

clientData	[In] The user-supplied data.
hOwner	[In] The parent window of the task pane view.
bShow	[In] It indicates whether the pane view is to be shown or to be hidden.

### Return

void

### Head file reference

fr\_appExpT.h: 4619

### Group

[FR\\_WinMSGCallbacksRec](#)

### Since

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRTTaskPaneViewCanClose

### Syntax

```
typedef FS_BOOL (*FRTTaskPaneViewCanClose)(  
    FS_LPVVOID clientData  
>);
```

### Description

It is called when the task pane view is to be closed.

### Parameter

clientData	[In] The user-supplied data.
------------	------------------------------

**Return**

TRUE if the task pane view can be closed, otherwise not.

**Head file reference**

fr\_appExpT.h: 4630

**Group**

[FR\\_WinMSGCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRTTaskPanelViewIsShowInMenu****Syntax**

```
typedef FS_BOOL (*FRTTaskPanelViewIsShowInMenu)(  
    FS\_LPVVOID clientData,  
    clientData,  
    void* pMsg  
);
```

**Description**

The callback is invoked by Foxit Reader to dispatch the message to plug-in.

[SDK\\_LATEEST\\_VERSION > 8.1](#)

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

clientData	[In] The user-supplied data.
------------	------------------------------

pMsg	[In] Points to a <i>MFC MSG</i> structure that contains the message to process.
------	---

**Return**

[TRUE](#) if the message was fully processed in [FRPIPreTranslateMessage](#) and should not be processed further. [FALSE](#) if the message should be processed in the normal way.

**Head file reference**

fr\_appExpT.h: 4669

**Group**

[FR\\_WinMSGCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 8.1](#)

**FRWndProviderCreateViewWnd****Syntax**

```
typedef void (*FRWndProviderCreateViewWnd)(  
    FS\_LPVVOID clientData,  
    FR\_Document frDoc,  
    HWND hParent  
)
```

**Description**

It is called by Foxit Reader to notify the plug-in to create the view.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The specified document.
-------	------------------------------

---

hParent	[In] The parent window.
---------	-------------------------

---

**Return**

void

**Head file reference**

[fr\\_appExpT.h: 5066](#)

**Group**

[FR\\_WndProviderCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRWndProviderOnHScroll****Syntax**

```
typedef void (*FRWndProviderOnHScroll)(  
    FS\_LPVVOID clientData,  
    FR\_Document frDoc,  
    unsigned int nSBCode,  
    unsigned int nPos,  
    void* pScrollBar  
)
```

**Description**

It is called by Foxit Reader when the user clicks a horizontally-aligned scroll bar, spin button control, or slider control.

**Parameter**

---

clientData	[In] The user-supplied data.
frDoc	[In] The specified document.
nSBCode	[In] Specifies a scroll-bar code that indicates the user's scrolling request. Reference to <i>MFC CWnd::OnHScroll</i> .
nPos	[In] Specifies the scroll-box position if the scroll-bar code is <i>SB_THUMBPOSITION</i> or <i>SB_THUMBTRACK</i> ; otherwise, not used. Depending on the initial scroll range, nPos may be negative and should be cast to an int if necessary.
pScrollBar	[In] It represents the MFC type of <i>CScrollBar</i> .

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 5084

**Group**[FR\\_WndProviderCallbacksRec](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FRWndProviderOnVScroll****Syntax**

```
typedef void (*FRWndProviderOnVScroll)(  
    FS\_LPVVOID clientData,  
    FR\_Document frDoc,  
    unsigned int nSBCode,  
    unsigned int nPos,  
    void* pScrollBar  
)
```

**Description**

It is called by Foxit Reader when the user clicks a vertically-aligned scroll bar, spin button control, or slider control.

**Parameter**


---

clientData	[In] The user-supplied data.
frDoc	[In] The specified document.

---

---

nSBCode	[In] Specifies a scroll-bar code that indicates the user's scrolling request. Reference to <i>MFC CWnd::OnHScroll</i> .
nPos	[In] Specifies the scroll-box position if the scroll-bar code is <i>SB_THUMBPOSITION</i> or <i>SB_THUMBTRACK</i> ; otherwise, not used. Depending on the initial scroll range, nPos may be negative and should be cast to an int if necessary.
pScrollBar	[In] It represents the MFC type of <i>CScrollBar</i> .

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 5100

**Group**[FR\\_WndProviderCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRWndProviderOnCmdMsg****Syntax**

```
typedef FS_BOOL (*FRWndProviderOnCmdMsg)(
    FS_LVOID clientData,
    FR_Document frDoc,
    unsigned int nID,
    FS_INT32 nCode,
    void* pExtra,
    void* pHandlerInfo
);
```

**Description**

It is called by the *Foxit Reader* to route and dispatch command messages and to handle the update of command user-interface objects, such as menu, toolbar.

**Parameter**


---

clientData	[In] The user-supplied data.
frDoc	[In] The specified document.
nID	[In] Contains the command ID.
nCode	[In] References to <i>MFC CCmdTarget::OnCmdMsg</i> .

---

---

pExtra	[In] References to <i>MFC CCmdTarget::OnCmdMsg</i> .
pHandlerInfo	[In] It represents the <i>MFC</i> struct <i>AFX_CMDHANDLERINFO</i> .

---

**Return**

Nonzero if the message is handled; otherwise 0.

**Head file reference**

fr\_appExpT.h: 5116

**Group**

[FR\\_WndProviderCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRWndProviderMoveWindow****Syntax**

```
typedef void (*FRWndProviderMoveWindow)(
    FS_LPVVOID clientData,
    FR_Document frDoc,
    FS_INT32 x,
    FS_INT32 y,
    FS_INT32 nWidth,
    FS_INT32 nHeight,
    FS_BOOL bRepaint
);
```

**Description**

It is called by the *Foxit Reader* to notify the plug-in to move the window.

**Parameter**


---

clientData	[In] The user-supplied data.
frDoc	[In] The specified document.
x	[In] Specifies the new position of the left side of the window.
y	[In] Specifies the new position of the top of the window.
nWidth	[In] Specifies the new width of the window.

---

---

nHeight	[In] Specifies the new height of the window.
---------	--

---

bRepaint	[In] Specifies whether window is to be repainted.
----------	---

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 5133

**Group**[FR\\_WndProviderCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRWndProviderOnSetCursor****Syntax**

```
typedef FS_BOOL (*FRWndProviderOnSetCursor)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc,  
    void* pWnd,  
    unsigned int nHitTest,  
    unsigned int message  
)
```

**Description**

It is called by the *Foxit Reader* to notify the plug-in to set the cursor. References to *MFC CWnd::OnSetCursor* for detailed description.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The specified document.
-------	------------------------------

---

pWnd	[In] It represents the <i>MFC CWnd*</i> and specifies a pointer to the window that contains the cursor. The pointer may be temporary and should not be stored for later use.
------	--

---

nHitTest	[In] Specifies the hit-test area code. The hit test determines the location of the cursor.
----------	--

---

message	[In] Specifies the mouse message number.
---------	--

---

**Return**

Nonzero to halt further processing, or 0 to continue.

**Head file reference**

fr\_appExpT.h: 5149

**Group**

[FR\\_WndProviderCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRWndProviderZoomToPage

**Syntax**

```
typedef void (*FRWndProviderZoomToPage)(  
    FS\_LPVOID clientData,  
    FR\_Document frDoc,  
    double dbScale,  
    FS\_BOOL bUpdate  
)
```

**Description**

It is called by the *Foxit Reader* to notify the plug-in to zoom the window.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The specified document.
-------	------------------------------

---

dbScale	[In] The specified zoom scale.
---------	--------------------------------

---

bUpdate	[In] Whether the window needs to update or not.
---------	---

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 5163

**Group**

[FR\\_WndProviderCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRWndProviderGotoPage

### Syntax

```
typedef void (*FRWndProviderGotoPage)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 nIndex  
)
```

### Description

It is called by the *Foxit Reader* to notify the plug-in to go to the specified page.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The specified document.
-------	------------------------------

---

nPageIndex	[In] The specified page index.
------------	--------------------------------

---

### Return

void

### Head file reference

fr\_appExpT.h: 5176

### Group

[FR\\_WndProviderCallbacksRec](#)

### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRWndProviderShowWindow

### Syntax

```
typedef void (*FRWndProviderShowWindow)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc,  
    FS_BOOL bShow  
)
```

### Description

It is called by the *Foxit Reader* to notify the plug-in to show the window created.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The specified document.
-------	------------------------------

---

bShow	[In] Whether the window is to be shown or to be hidden.
-------	---

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 5189

**Group**[FR\\_WndProviderCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRWndProviderReInitScrollBar****Syntax**

```
typedef void (*FRWndProviderReInitScrollBar)(  
    FS\_LVOID clientData,  
    FR\_Document frDoc  
)
```

**Description**

This interface is reserved.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The specified document.
-------	------------------------------

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 5201

**Group**[FR\\_WndProviderCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRWndProviderGetPageIndex

### Syntax

```
typedef FS_INT32 (*FRWndProviderGetPageIndex)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc  
);
```

### Description

This interface is reserved.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

frDoc	[In] The specified document.
-------	------------------------------

---

### Return

void

### Head file reference

fr\_appExpT.h: 5213

### Group

[FR\\_WndProviderCallbacksRec](#)

### Since

[SDK LATEEST VERSION > 2.0](#)

## FRWndProviderInitScrollBar

### Syntax

```
typedef void (*FRWndProviderInitScrollBar)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc,  
    void* pHScroll,  
    void* pVScroll  
);
```

### Description

It is called by the *Foxit Reader* to notify the plug-in to init the scroll bar.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

frDoc	[In] The specified document.
pHScroll	[In] The pointer to the horizon scroll bar. It represents the MFC type of <i>CScrollBar</i> .
pVScroll	[In] The pointer to the vertical scroll bar. It represents the MFC type of <i>CScrollBar</i> .

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 5227

**Group**[FR\\_WndProviderCallbacksRec](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FRWndProviderOnSetFocus****Syntax**

```
typedef void (*FRWndProviderOnSetFocus)(  
    FS_LPVVOID clientData,  
    FR_Document frDoc,  
    void* pOldWnd  
)
```

**Description**It is called by the *Foxit Reader* after the window gains the input focus.**Parameter**


---

clientData	[In] The user-supplied data.
frDoc	[In] The specified document.

---

pOldWnd	[In] Contains the CWnd object that loses the input focus (may be NULL). The pointer may be temporary and should not be stored for later use.
---------	--

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 5240

**Group**[FR\\_WndProviderCallbacksRec](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FRWndProviderOnMouseWheel****Syntax**

```
typedef FS_BOOL (*FRWndProviderOnMouseWheel)(  
    FS\_LPVOID clientData,  
    FR\_Document frDoc,  
    unsigned int nFlags,  
    short zDelta,  
    FS\_INT32 x,  
    FS\_INT32 y  
)
```

**Description**

It is called by the *Foxit Reader* as a user rotates the mouse wheel and encounters the wheel's next notch.

**Parameter**


---

clientData	[In] The user-supplied data.
frDoc	[In] The specified document.
nFlags	[In] Indicates whether various virtual keys are down. References to <i>MFC CWnd::OnMouseWheel</i> .
zDelta	[In] Indicates distance rotated. References to <i>MFC CWnd::OnMouseWheel</i> .
x	[In] Specifies the x-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the screen.
y	[In] Specifies the y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the screen.

---

**Return**

Nonzero if mouse wheel scrolling is enabled; otherwise 0.

**Head file reference**

[fr\\_appExpT.h: 5256](#)

**Group**[FR\\_WndProviderCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRWndProviderOnRelease****Syntax**

```
typedef void (*FRWndProviderOnRelease)(  
    FS_LPVOID clientData  
)
```

**Description**

It is called by the *Foxit Reader* to notify the plug-in to release the resource.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 5267

**Group**[FR\\_WndProviderCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)

## Structures

### Structures summary

**[FR\\_MSGBOX\\_CHECKBOXPARAMS](#)**

A data structure representing the params of the check box on the message box.

**[FR\\_SignatureBaseInfo](#)**

[FR\\_SignatureBaseInfo](#) represents the base info of the signature, including the time, the name, the state and so on. You can get the base info of the signature through [FRCustomSignatureGetSignatureBaseInfo](#).

**[FR\\_SignatureDictInfo](#)**

Data structure for signature dictionary info.

**[FR\\_SignatureInfo](#)**

You have to fill the signature info when you want to sign the PDF.

**[FR\\_SignaturePosInfo](#)**

You have to fill the signature position info when you want to sign the PDF.

#### [FR\\_SignatureTimestamp](#)

The definition of the signature timestamp.

#### [FR\\_SignatureTimestampServer](#)

The definition of the signature timestamp server info.

#### [FR\\_WINDIB\\_Open\\_Args](#)

A structure used to indicate how to open a new image file or stream.

#### [FRBULBMESSAGEINFO](#)

A data structure representing a bulb message information.

#### [FRBULBMESSAGEINFO2](#)

A data structure of second version representing a bulb message information. A new property of message title is added to the data structure. See [FRBulbMsgCenterAddMessage3](#) and [FRBulbMsgCenterAddMessage4](#).

## Structs detail

### [FR\\_MSGBOX\\_CHECKBOXPARAMS](#)

#### **Syntax**

```
typedef struct __fr_msgbox_checkboxparams__{
    FS_BOOL bShowCheckBox,
    FS_LPCWSTR wszCheckBoxText,
    FS_BOOL bIsChecked
}FR_MSGBOX_CHECKBOXPARAMS, *PFR_MSGBOX_CHECKBOXPARAMS;
```

#### **Description**

A data structure representing the params of the check box on the message box.

#### **Head file reference**

fr\_sysExpT.h: 54

#### **bShowCheckBox**

Whether show the check box or not.

#### **wszCheckBoxText**

The text of the check box.

#### **bIsChecked**

Whether the check box is checked or not.

### [FR\\_SignatureBaseInfo](#)

#### **Syntax**

```
typedef struct __FR_SignatureBaseInfo__{
    memset(wsSignedAutorName, 0, 128*sizeof(wchar_t)),
    memset(wsSignatureName, 0, 128*sizeof(wchar_t)),
    bSignedField = FALSE,
    nSignedPageIndex = 0,
    bVerified = FALSE,
    dwVerifyState = 0,
    FILETIME ftSignedTime,
    FILETIME ftValidTimeNotBefore,
    FILETIME ftValidTimeNotAfter,
```

---

```
wchar_t wsSignedAutorName[128],  
wchar_t wsSignatureName[128],  
FS_BOOL bSignedField,  
FS_INT32 nSignedPageIndex,  
FS_DWORD dwVerifyState,  
FS_BOOL bVerified  
}FR_SignatureBaseInfo;
```

**Description**

[FR\\_SignatureBaseInfo](#) represents the base info of the signature, including the time, the name, the state and so on. You can get the base info of the signature through [FRCustomSignatureGetSignatureBaseInfo](#) .

**Head file reference**

fr\_docExpT.h: 662

**0, 128\*sizeof(wchar\_t))**

[FR\\_SignatureBaseInfo](#) represents the base info of the signature, including the time, the name, the state and so on. You can get the base info of the signature through [FRCustomSignatureGetSignatureBaseInfo](#) .

**0, 128\*sizeof(wchar\_t))**

[FR\\_SignatureBaseInfo](#) represents the base info of the signature, including the time, the name, the state and so on. You can get the base info of the signature through [FRCustomSignatureGetSignatureBaseInfo](#) .

**= FALSE**

[FR\\_SignatureBaseInfo](#) represents the base info of the signature, including the time, the name, the state and so on. You can get the base info of the signature through [FRCustomSignatureGetSignatureBaseInfo](#) .

**= 0**

[FR\\_SignatureBaseInfo](#) represents the base info of the signature, including the time, the name, the state and so on. You can get the base info of the signature through [FRCustomSignatureGetSignatureBaseInfo](#) .

**= FALSE**

[FR\\_SignatureBaseInfo](#) represents the base info of the signature, including the time, the name, the state and so on. You can get the base info of the signature through [FRCustomSignatureGetSignatureBaseInfo](#) .

**= 0**

[FR\\_SignatureBaseInfo](#) represents the base info of the signature, including the time, the name, the state and so on. You can get the base info of the signature through [FRCustomSignatureGetSignatureBaseInfo](#) .

**ftSignedTime**

The time when the signature was signed.

**ftValidTimeNotBefore**

The time before which the signature is invalid.

**ftValidTimeNotAfter**

The time after which the signature is invalid.

**wsSignedAutorName[128]**

The author name.

**wsSignatureName[128]**

The signature name.

**bSignedField**

Whether it is a signed field or not.

**nSignedPageIndex**

The index of the page where the signature was signed.

**dwVerifyState**

The verifying state.

**bVerified**

Whether the signature has been verified or not.

**FR\_SignatureDictInfo****Syntax**

```
typedef struct __FR_SignatureDictInfo__{
    memset(wsName, 0, 128*sizeof(wchar_t)),
    memset(wsDate, 0, 128*sizeof(wchar_t)),
    memset(wsReason, 0, 256*sizeof(wchar_t)),
    memset(wsLocation, 0, 256*sizeof(wchar_t)),
    memset(wsFilter, 0, 128*sizeof(wchar_t)),
    wchar_t wsName[128],
    wchar_t wsDate[128],
    wchar_t wsReason[256],
    wchar_t wsLocation[256],
    wchar_t wsFilter[128]
}FR_SignatureDictInfo;
```

**Description**

Data structure for signature dictionary info.

**Head file reference**

fr\_docExpT.h: 291

**0, 128\*sizeof(wchar\_t))**

Data structure for signature dictionary info.

**0, 128\*sizeof(wchar\_t))**

Data structure for signature dictionary info.

**0, 256\*sizeof(wchar\_t))**

Data structure for signature dictionary info.

**0, 256\*sizeof(wchar\_t))**

Data structure for signature dictionary info.

**0, 128\*sizeof(wchar\_t))**

Data structure for signature dictionary info.

**wsName[128]**

The signer.

**wsDate[128]**

The signing date.

**wsReason[256]**

The reason for signing.

**wsLocation[256]**

The location for signing.

**wsFilter[128]**

The sub-filter of the signing.

**FR\_SignatureInfo****Syntax**

```
typedef struct __FR_SignatureInfo__{
    memset(wsCN, 0, 64*sizeof(FS_WCHAR)),
    memset(wsO, 0, 64*sizeof(FS_WCHAR)),
    memset(wsOU, 0, 64*sizeof(FS_WCHAR)),
    memset(wsE, 0, 128*sizeof(FS_WCHAR)),
    memset(wsC, 0, 64*sizeof(FS_WCHAR)),
    lShowFlag = FR_SIG_SHOW_ALL,
    TextDir = FR_SGTD_AUTO,
    Icontype = FR_SGIT_NAME,
    memset(ImagePath, 0, 260),
    nImageOpt = 100,
    PermissionType = FR_APG_NONE,
    lEstimateSignData = 0,
    bDocTimeStamp = false,
    FS_WCHAR wsCN[64],
    FS_WCHAR wsO[64],
    FS_WCHAR wsOU[64],
    FS_WCHAR wsE[128],
    FS_WCHAR wsC[64],
    FR_SignatureDictInfo SignDictInfo,
    unsigned long lShowFlag,
    FR_SG_ICONTYPE Icontype,
    FR_SG_TEXTDIR TextDir,
    FR_SG_PERMISSION PermissionType,
    FS_WCHAR ImagePath[260],
    FS_INT32 nImageOpt,
    unsigned long lEstimateSignData,
    FS_BOOL bDocTimeStamp
}FR_SignatureInfo;
```

**Description**

You have to fill the signature info when you want to sign the PDF.

**Head file reference**

fr\_docExpT.h: 363

**0, 64\*sizeof(FS\_WCHAR))**

You have to fill the signature info when you want to sign the PDF.

**0, 64\*sizeof(FS\_WCHAR))**

You have to fill the signature info when you want to sign the PDF.

**0, 64\*sizeof(FS\_WCHAR))**

You have to fill the signature info when you want to sign the PDF.

**0, 128\*sizeof(FS\_WCHAR))**

You have to fill the signature info when you want to sign the PDF.

**0, 64\*sizeof(FS\_WCHAR))**

You have to fill the signature info when you want to sign the PDF.

**= FR\_SIG\_SHOW\_ALL**

You have to fill the signature info when you want to sign the PDF.

**= FR\_SGTD\_AUTO**

You have to fill the signature info when you want to sign the PDF.

**= FR\_SGIT\_NAME**

You have to fill the signature info when you want to sign the PDF.

**0, 260)**

You have to fill the signature info when you want to sign the PDF.

**= 100**

You have to fill the signature info when you want to sign the PDF.

**= FR\_APG\_NONE**

You have to fill the signature info when you want to sign the PDF.

**= 0**

You have to fill the signature info when you want to sign the PDF.

**= false**

You have to fill the signature info when you want to sign the PDF.

**wsCN[64]**

**wsO[64]**

**wsOU[64]**

**wsE[128]**

**wsC[64]****SignDictInfo****long lShowFlag**The flag for showing appearance. See [FRSIGShowAPFlags](#).**Icontype****TextDir****PermissionType****ImagePath[260]****nImageOpt**

The image opacity.

**long lEstimateSignData****bDocTimeStamp****FR\_SignaturePosInfo****Syntax**

```
typedef struct __FR_SignaturePosInfo__{
    pPDFDoc = NULL,
    nPageIndex = -1,
    nLeft = nRight = nTop = nBottom = 0,
    memset(FileSavePath, 0, 260 * sizeof(FS_WCHAR)),
    FPD_Document pPDFDoc,
    FS_INT32 nIndex,
    FS_INT32 nLeft,
    FS_INT32 nRight,
    FS_INT32 nTop,
    FS_INT32 nBottom,
    FS_WCHAR FileSavePath[260]
}FR_SignaturePosInfo;
```

**Description**

You have to fill the signature position info when you want to sign the PDF.

**Head file reference**

fr\_docExpT.h: 479

**= NULL**

You have to fill the signature position info when you want to sign the PDF.

**= -1**

You have to fill the signature position info when you want to sign the PDF.

**= nRight = nTop = nBottom = 0**

You have to fill the signature position info when you want to sign the PDF.

**0, 260 \* sizeof(FS\_WCHAR)**

You have to fill the signature position info when you want to sign the PDF.

**pPDFDoc**

You have to fill the signature position info when you want to sign the PDF.

**nPageIndex**

You have to fill the signature position info when you want to sign the PDF.

**nLeft**

You have to fill the signature position info when you want to sign the PDF.

**nRight**

You have to fill the signature position info when you want to sign the PDF.

**nTop**

You have to fill the signature position info when you want to sign the PDF.

**nBottom**

You have to fill the signature position info when you want to sign the PDF.

**FileSavePath[260]**

You have to fill the signature position info when you want to sign the PDF.

**FR\_SignatureTimestamp****Syntax**

```
typedef struct __FR_SignatureTimestamp__{  
    dwLowDateTime = 0,  
    dwHighDateTime = 0,  
    FS_DWORD dwLowDateTime,  
    FS_DWORD dwHighDateTime  
}FR_SignatureTimestamp;
```

**Description**

The definition of the signature timestamp.

**Head file reference**

fr\_docExpT.h: 428

**= 0**

The definition of the signature timestamp.

**= 0**

The definition of the signature timestamp.

**dwLowDateTime****dwHighDateTime**

## FR\_SignatureTimestampServer

### Syntax

```
typedef struct __FR_SignatureTimestampServer__{
    memset(wsServerURL, 0, 2048 * sizeof(FS_WCHAR)),
    memset(wsUserName, 0, 128 * sizeof(FS_WCHAR)),
    memset(wsPassWord, 0, 128 * sizeof(FS_WCHAR)),
    FS_WCHAR wsServerURL[2048],
    FS_WCHAR wsUserName[128],
    FS_WCHAR wsPassWord[128]
}FR_SignatureTimestampServer;
```

### Description

The definition of the signature timestamp server info.

#### Head file reference

fr\_docExpT.h: 452

##### **0, 2048 \* sizeof(FS\_WCHAR)**

The definition of the signature timestamp server info.

##### **0, 128 \* sizeof(FS\_WCHAR)**

The definition of the signature timestamp server info.

##### **0, 128 \* sizeof(FS\_WCHAR)**

The definition of the signature timestamp server info.

##### **wsServerURL[2048]**

The server URL.

##### **wsUserName[128]**

The user name.

##### **wsPassWord[128]**

The password.

## FR\_WINDIB\_Open\_Args

### Syntax

```
typedef struct _FR_WINDIB_Open_Args__{
    FS_INT32 flags,
    const FS_BYTE* memory_base,
    FS_DWORD memory_size,
    FS_LPCWSTR path_name
}FR_WINDIB_Open_Args;
```

### Description

A structure used to indicate how to open a new image file or stream.

#### Head file reference

fr\_sysExpT.h: 68

**flags**

A set of bit flags indicating how to use the structure.

**FS\_BYTE\* memory\_base**

The first byte of the file in memory.

**memory\_size**

The size in bytes of the file in memory.

**path\_name**

A pointer to an 16-bit file pathname.

**FRBULBMESSAGEINFO****Syntax**

```
typedef struct __FRBULBMESSAGEINFO__{
    void* pClientData,
    FS_LPCSTR lpsMsgName,
    FS_LPCWSTR lpwsMessage,
    FRMSGIMPORTANCE importance,
    FS_BOOL bShowCheckBox,
    FS_BOOL bCheck,
    FS_LPCWSTR lpwsCheckBoxTitle,
    FRBulbMsgCheckCallback ckCallbackFunc,
    FS_INT32 opBtnCount,
    FS_ByteStringArray opBtnNames,
    FS_WideStringArray opBtnTitles,
    FRBulbMsgOperationCallback* opCallbackFuncList,
    FS_BOOL bCanBeCleared,
    FS_BOOL bShowDoNotAgainBtn,
    FS_BOOL bSpecial,
    pClientData = NULL,
    lpsMsgName = "",
    lpwsMessage = (FS_LPCWSTR)L"",
    lpwsCheckBoxTitle = (FS_LPCWSTR)L"",
    importance = FR_IMPO_LOW,
    bShowCheckBox = FALSE,
    bCheck = FALSE,
    ckCallbackFunc = NULL,
    opBtnCount = 0,
    opBtnNames = NULL,
    opBtnTitles = NULL,
    opCallbackFuncList = NULL,
    bCanBeCleared = TRUE,
    bShowDoNotAgainBtn = FALSE,
    bSpecial = FALSE
}FRBULBMESSAGEINFO;
```

**Description**

A data structure representing a bulb message information.

**Head file reference**

fr\_barExpT.h: 1567

**pClientData**

The user-supplied data.

**lpsMsgName**

The unique name of the bulb message. Required.

**lpwsMessage**

The message content. Required.

**importance**

The importance type of the bulb message. The default value is [FR\\_IMPO\\_LOW](#).

**bShowCheckBox**

Whether to show the check box of "Don't show again."

**bCheck**

Whether to check the check box of "Don't show again."

**lpwsCheckBoxTitle**

The title of the check box."

**ckCallbackFunc**

The callback is invoked by Foxit Reader when user clicks the check box of the bulb message.

**opBtnCount**

The count of the operation buttons.

**opBtnNames**

The byte string array of operation button names. The array size must be equal to opBtnCount.

**opBtnTitles**

The wide string array of operation button titles. The array size must be equal to opBtnCount.

**opCallbackFuncList**

The callbacks are invoked by Foxit Reader when user clicks the operation buttons of the bulb message.

**bCanBeCleared**

Indicates whether the bulb message can be cleared.

**bShowDoNotAgainBtn**

Indicates whether the Don't show again button can be show.

**bSpecial**

Indicates whether the Don't show again button can be show.

**= NULL**

Indicates whether the Don't show again button can be show.

= ""

Indicates whether the Don't show again button can be show.

= **(FS\_LPCWSTR)L""**

Indicates whether the Don't show again button can be show.

= **(FS\_LPCWSTR)L""**

Indicates whether the Don't show again button can be show.

= **FR\_IMPO\_LOW**

Indicates whether the Don't show again button can be show.

= **FALSE**

Indicates whether the Don't show again button can be show.

= **FALSE**

Indicates whether the Don't show again button can be show.

= **NULL**

Indicates whether the Don't show again button can be show.

= **0**

Indicates whether the Don't show again button can be show.

= **NULL**

Indicates whether the Don't show again button can be show.

= **NULL**

Indicates whether the Don't show again button can be show.

= **NULL**

Indicates whether the Don't show again button can be show.

= **TRUE**

Indicates whether the Don't show again button can be show.

= **FALSE**

Indicates whether the Don't show again button can be show.

= **FALSE**

Indicates whether the Don't show again button can be show.

## FRBULBMESSAGEINFO2

### Syntax

```
typedef struct __FRBULBMESSAGEINFO2__{
    unsigned long lStructSize,
    void* pClientData,
    FRBULBMESSAGEINFO bulbMsgInfoV1,
    FS\_LPCWSTR lpwsTitle,
```

---

```
    pClientData = NULL,
    lpwsTitle = (FS_LPCWSTR)L"""
}FRBULBMESSAGEINFO2;
```

**Description**

A data structure of second version representing a bulb message information. A new property of message title is added to the data structure. See [FRBulbMsgCenterAddMessage3](#) and [FRBulbMsgCenterAddMessage4](#).

**Head file reference**

fr\_barExpT.h: 1642

**long lStructSize**

The size of data structure. It must be set to *sizeof(FRBULBMESSAGEINFO2)*.

**pClientData**

The user-supplied data.

**bulbMsgInfoV1**

The user-supplied data.

**lpwsTitle**

The message title.

**= NULL**

The message title.

**= (FS\_LPCWSTR)L""**

The message title.

## FDRM\_CategoryRead

**Used by****Description****Used by**

[FDRMCategoryReadCountSubCategories](#)  
[FDRMCategoryReadDestroy](#)

**Functions****Functions summary**

[FDRMCategoryReadCountSubCategories](#)  
[FDRMCategoryReadDestroy](#)

**Functions detail**

## FDRMCategoriesReadCountSubCategories

### Syntax

```
FS_INT32 FDRMCategoriesReadCountSubCategories (
    FDRM\_CategoryRead reader,
    FDRM\_CATEGORY\_HANDLER hParent,
    FS\_ByteString bsFilter
);
```

### Description

#### Parameter

---

reader	[In]
--------	------

---

hParent	[In]
---------	------

---

bsFilter	[In]
----------	------

---

#### Return

#### Head file reference

[fdm\\_descTempl.h](#): 22

#### Since

[SDK LATEST VERSION > 1.0](#)

## FDRMCategoriesReadDestroy

### Syntax

```
void FDRMCategoriesReadDestroy (
    FDRM\_CategoryRead reader,
    FDRM\_CATEGORY\_HANDLER hParent,
    FS\_ByteString bsFilter,
    FS\_INT32 index
);
```

### Description

#### Parameter

---

reader	[In]
--------	------

---

hParent	[In]
---------	------

---

bsFilter	[In]
----------	------

---

[index](#)

[In]

---

**Return**

**Head file reference**

fdrm\_descTempl.h: 60

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## **FDRM\_CategoryWrite**

[Description](#)

## **FDRM\_DescData**

[Description](#)

## **FDRM\_DescRead**

[Description](#)

## **FDRM\_DescWrite**

[Description](#)

## **FDRM\_EncryptDictRead**

[Description](#)

## **FDRM\_Encryptor**

[Description](#)

## **FDRM\_EnvelopeRead**

[Description](#)

## **FDRM\_FoacRead**

[Description](#)

## **FDRM\_FoacWrite**

Description

## **FDRM\_Mgr**

Description

## **FDRM\_PDFSchema**

Description

## **FDRM\_PKI**

Description

## **FDRM\_PresentationData**

Description

## **FDRM\_ScriptData**

Description

## **FDRM\_SignatureData**

Description

## **FOFD\_Action**

Description

## **FOFD\_Actions**

Description

## **FOFD\_CodeC**

Description

## **FOFD\_Creator**

Description

## **FOFD\_CryptoDict**

Description

## **FOFD\_Dest**

Description

## **FOFD\_Doc**

Description

## **FOFD\_Outline**

Description

## **FOFD\_Package**

Description

## **FOFD\_Page**

Description

## **FOFD\_Parser**

Description

## **FOFD\_Perms**

Description

## **FOFD\_PrintSetting**

Description

## **FOFD\_Sign**

Description

## **FOFD\_SM4CryptoHandler**

Description

## **FOFD\_SMSecurityHandler**

Description

## **FOFD\_StdCertSecurityHandler**

Description

## **FOFD\_StdCryptoHandler**

Description

## **FOFD\_StdSecurityHandler**

Description

## **FOFD\_Sys**

Description

## **FOFD\_UIMgr**

Description

## **FOFD\_VPrefers**

Description

## **FOFD\_WriteDoc**

Description

## **FR\_ActionWizardData**

Description

## **FR\_ActionWizardHandler**

Description

Definitions

Definitions summary

[\*\*FR\\_ACTIONWIZARD\\_NOCREATE\*\*](#)  
[\*\*FR\\_ACTIONWIZARD\\_NOPRESET\*\*](#)

**FR\_ACTIONWIZARD\_PRESET**

**FR\_ACTIONWIZARD\_PROMPTUSER**

**Definitions detail**

**FR\_ACTIONWIZARD\_NOCREATE**

**Syntax**

```
#define FR_ACTIONWIZARD_NOCREATE 0x004
```

**Description**

**Group**

[ActionWizardPresetFlag](#)

**Head file reference**

fr\_appExpT.h: 7448

**FR\_ACTIONWIZARD\_NOPRESET**

**Syntax**

```
#define FR_ACTIONWIZARD_NOPRESET 0x000
```

**Description**

**Group**

[ActionWizardPresetFlag](#)

**Head file reference**

fr\_appExpT.h: 7442

**FR\_ACTIONWIZARD\_PRESET**

**Syntax**

```
#define FR_ACTIONWIZARD_PRESET 0x001
```

**Description**

**Group**

[ActionWizardPresetFlag](#)

**Head file reference**

fr\_appExpT.h: 7444

**FR\_ACTIONWIZARD\_PROMPTUSER**

**Syntax**

```
#define FR_ACTIONWIZARD_PROMPTUSER 0x002
```

**Description****Group**[ActionWizardPresetFlag](#)**Head file reference**

fr\_appExpT.h: 7446

**Enumerations****Enumerations summary**[\*\*FRACTIONWIZARDEXECUTESTATUS\*\*](#)**Enumerations detail****FRACTIONWIZARDEXECUTESTATUS****Syntax**

```
enum FRACTIONWIZARDEXECUTESTATUS{
    FR_EXECUTE_STATUS_DONTSUPPORT,
    FR_EXECUTE_STATUS_FAILED,
    FR_EXECUTE_STATUS_SUCESS,
    FR_EXECUTE_STATUS_CANCEL,
    FR_EXECUTE_STATUS_SHOWPROMPT /**. */  
};
```

**Description****Head file reference**

fr\_appExpT.h: 7425

**FR\_EXECUTE\_STATUS\_DONTSUPPORT****FR\_EXECUTE\_STATUS\_FAILED****FR\_EXECUTE\_STATUS\_SUCESS****FR\_EXECUTE\_STATUS\_CANCEL****FR\_EXECUTE\_STATUS\_SHOWPROMPT /\*\*. \*/****Callbacks**

## Callbacks summary

### Callbacks detail

## FR\_Annot

### [Return from Used by](#)

#### Description

The UI layer annotation object.

#### Returned from

[FRPageViewAddAnnot](#)  
[FRPageViewGetAnnotAtPoint](#)  
[FRPageViewGetAnnotByIndex](#)  
[FRPageViewGetFocusAnnot](#)

#### Used by

[FRDocIsValidAnnot](#)  
[FRDocSetFocusAnnot](#)  
[FRPageViewDeleteAnnot](#)  
[FRPageViewUpdateAllViews](#)  
[FRAnnotGetPageView](#)  
[FRAnnotGetPDFAnnot](#)  
[FRAnnotGetSubType](#)  
[FRAnnotGetType](#)  
[FRAnnotSetVisible](#)

## Functions

### Functions summary

#### [FRAnnotGetPageView](#)

Gets the associated page view.

#### [FRAnnotGetPDFAnnot](#)

Gets the data layer PDF annotation.

#### [FRAnnotGetSubType](#)

Gets the sub type of the annotation.

#### [FRAnnotGetType](#)

Gets the type of the annotation.

#### [FRAnnotSetVisible](#)

Sets the annotation to be visible or not.

### Functions detail

#### FRAnnotGetPageView

##### Syntax

FR\_PageView FRAnnotGetPageView (

```
FR_Annot frAnnot,  
FS_INT32 nIndex  
);
```

**Description**

Gets the associated page view.

**Parameter**

---

frAnnot	[In] The input UI layer annotation object.
---------	--

---

nIndex	[In] The specified page view index.
--------	-------------------------------------

---

**Return**

The associated page view.

**Head file reference**

fr\_viewTempl.h: 1062

**Since**

[SDK\\_LATEEST\\_VERSION > 8.1](#)

**FRAnnotGetPDFAnnot****Syntax**

```
FPD_Annot FRAnnotGetPDFAnnot (  
    FR_Annot frAnnot  
);
```

**Description**

Gets the data layer PDF annotation.

**Parameter**

---

frAnnot	[In] The input UI layer annotation object.
---------	--

---

**Return**

The data layer PDF annotation.

**Head file reference**

fr\_viewTempl.h: 1019

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRAnnotGetSubType**

**Syntax**

```
void FRAnnotGetSubType (
    FR\_Annot frAnnot,
    FS\_ByteString* outSubType
);
```

**Description**

Gets the sub type of the annotation.

**Parameter**

---

frAnnot	[In] The input UI layer annotation object.
---------	--

---

outSubType	[Out] It receives the sub type of the annotation.
------------	---

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 1040

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRAnnotGetType****Syntax**

```
void FRAnnotGetType (
    FR\_Annot frAnnot,
    FS\_ByteString* outType
);
```

**Description**

Gets the type of the annotation.

**Parameter**

---

frAnnot	[In] The input UI layer annotation object.
---------	--

---

outType	[Out] It receives the type of the annotation.
---------	---

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 1029

**Since**[SDK LATEEST VERSION > 1.0](#)

## FRAnnotSetVisible

**Syntax**

```
void FRAnnotSetVisible (
    FR\_Annot frAnnot,
    FS\_BOOL bShow
);
```

**Description**

Sets the annotation to be visible or not.

**Parameter**

---

frAnnot	[In] The input UI layer annotation object.
---------	--

---

bShow	[In] It indicates whether the annotation is visible.
-------	--

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 1051

**Note:** You must call this interface in FRPageViewOnWillParsePage callback.

**Since**[SDK LATEEST VERSION > 2.1](#)

## FR\_App

**Used by****Description**

[FR\\_App](#) represents the Foxit Reader application itself. From the application layer, you can control the appearance of Foxit Reader, whether Reader appears, and the size of the application window. Your application has access to the menu bar and the toolbar through this object. The application layer also provides access to the visual presentation of a PDF file on the screen (an [FR\\_Document](#) ).

**Used by**[FRAppRegisterAppEventHandler](#)

## Enumerations

### Enumerations summary

#### [FRDECMFileDialogType](#)

ECM Save type. See [FRAppFileDialogECMSaveProc](#).

### Enumerations detail

#### FRDECMFileDialogType

##### Syntax

```
enum FRDECMFileDialogType{
    FR\_ECMFileDialog\_Open\_Default,
    FR\_ECMFileDialog\_Open\_GetFileOnly,
    FR\_ECMFileDialog\_Save\_Default,
    FR\_ECMFileDialog\_Save\_ExportOffice,
    FR\_ECMFileDialog\_Save\_ExportBitmap,
    FR\_ECMFileDialog\_Save\_ExportOther
};
```

##### Description

ECM Save type. See [FRAppFileDialogECMSaveProc](#).

##### Head file reference

fr\_appExpT.h: 84

[FR\\_ECMFileDialog\\_Open\\_Default](#)

[FR\\_ECMFileDialog\\_Open\\_GetFileOnly](#)

[FR\\_ECMFileDialog\\_Save\\_Default](#)

[FR\\_ECMFileDialog\\_Save\\_ExportOffice](#)

[FR\\_ECMFileDialog\\_Save\\_ExportBitmap](#)

[FR\\_ECMFileDialog\\_Save\\_ExportOther](#)

## Callbacks

### Callbacks summary

#### [FRAppCustomRecentFileExecuteProc](#)

When the custom recent file item is clicked, the callback will be invoked. If it is not set, the default open process will be invoked.

#### [FRAppFileDialogECMOpenProc](#)

When the filedialog ECM item selected, the callback will be invoked.

#### [FRAppFileDialogECMSaveProc](#)

When the filedialog ECM item selected, the callback will be invoked.

#### [FRAppUndoRedoExitEditProc](#)

The callback will be invoked when you exit the editing mode. Then you can release the data.

## Callbacks detail

### FRApCustomRecentFileExecuteProc

#### Syntax

```
typedef FS_BOOL (*FRApCustomRecentFileExecuteProc)(  
    FS_LPCWSTR lpwsFile,  
    void clientData  
)
```

#### Description

When the custom recent file item is clicked, the callback will be invoked. If it is not set, the default open process will be invoked.

#### Parameter

---

lpwsFile	[In] The file path of the custom recent file item.
clientData	[In] The client data passed through <a href="#">FRApAddFileToCustomRecentFileList</a>

---

#### Return

[TRUE](#) if you handle the open process successfully in the callback, otherwise not.

#### Head file reference

fr\_appExpT.h: 117

#### Related method

[FRApAddFileToCustomRecentFileList](#)

#### Since

[SDK LATEEST VERSION > 1.0](#)

### FRApFileDialogECMOpenProc

#### Syntax

```
typedef FS_BOOL (*FRApFileDialogECMOpenProc)(  
    FRDECMFileDialogType type,  
    FS_LPCWSTR lpwPluginName,  
    FS_LPCWSTR lpwPluginHftName,  
    FS_LPCWSTR wsFilter,  
    void* clientData,  
    FS_LPCWSTR& lpwsFilePaths  
)
```

#### Description

When the filedialog ECM item selected, the callback will be invoked.

#### Parameter

---

type	[In] The specified file dialog type.
------	--------------------------------------

---

lpwPluginName	[In] The plugin name.
---------------	-----------------------

---

lpwPluginHftName	[In] The plugin HFT name.
------------------	---------------------------

---

wsFilter	[In] The specified filter.
----------	----------------------------

---

clientData	[In] The client data passed through <a href="#">FRAppAddECMFileDialog</a>
------------	---

---

lpwsFilePaths	[Out] The files selected by the ECM plugin, with " " as the boundary between the files.
---------------	---

---

#### Return

[TRUE](#) if you handle the filedialog process successfully in the callback, otherwise not.

#### Head file reference

fr\_appExpT.h: 135

#### Related method

[FRAppAddECMFileDialog](#)

## FRApFileDialogECMSaveProc

#### Syntax

```
typedef FS_BOOL (*FRApFileDialogECMSaveProc)(  
    FS_LPCWSTR wsSrcFilePath,  
    FS_LPCWSTR lpwPluginName,  
    FS_LPCWSTR lpwPluginHftName,  
    FRDECMFileDialogType type,  
    FS_LPCWSTR wsFilter,  
    void* clientData  
>;
```

#### Description

When the filedialog ECM item selected, the callback will be invoked.

#### Parameter

---

wsSrcFilePath	[In] The source file path.
---------------	----------------------------

---

---

lpwPluginName	[In] The plugin name
lpwPluginHftName	[In] The plugin HFT name
type	[In] The specified file dialog type.
wsFilter	[In] The specified filter.
clientData	[In] The client data passed through <a href="#">FRAppAddECMFileDialog</a>

---

**Return**

[TRUE](#) if you handle the filedialog process successfully in the callback, otherwise not.

**Head file reference**

fr\_appExpT.h: 152

**Related method**

[FRAppAddECMFileDialog](#)

**FRAppUndoRedoExitEditProc****Syntax**

```
typedef void (*FRAppUndoRedoExitEditProc)(
    void clientData
);
```

**Description**

The callback will be invoked when you exit the editing mode. Then you can release the data.

**Parameter**


---

clientData	[In] The client data passed through <a href="#">FRAppUndoRedoBeginEdit</a>
------------	--

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 104

**Related method**

[FRAppUndoRedoBeginEdit](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## Functions

### Functions summary

#### [\*\*FRAppActiveChildFrame\*\*](#)

Activates the specified child frame.

#### [\*\*FRAppAddDocPropertyPage\*\*](#)

Adds a document property page into Reader document properties dialog.

#### [\*\*FRAppAddFileAttachment\*\*](#)

Add a embed file to current document with The file path.

#### [\*\*FRAppAddFileToCustomRecentFileList\*\*](#)

Adds the recent file to the custom recent file list.

#### [\*\*FRAppAddFileToCustomRecentFileList2\*\*](#)

Adds the recent file to the custom recent file list. Registers the Ribbon recent file event handler to catch the pinning and removing event, so that the plug-in can update the data layer.

#### [\*\*FRAppAddFileToRecentFileList\*\*](#)

Adds the recent opened file path to the list.

#### [\*\*FRAppAddLog\*\*](#)

Outputs log information.

#### [\*\*FRAppAddNavPanelView\*\*](#)

Adds a navigation panel view into the Foxit Reader. The navigation panel view can be added after reader has started up.

#### [\*\*FRAppAddPreferencePage\*\*](#)

Adds a preference page into Reader preference dialog.

#### [\*\*FRAppAddUndoRedoItem\*\*](#)

Adds a undo-redo item to the specified document.

#### [\*\*FRAppAdvWndSetCustomize\*\*](#)

Sets whether to customize the advertisement.

#### [\*\*FRAppCanQuit\*\*](#)

Gets the permission to decide whether you can quit.

#### [\*\*FRAppCheckModuleLicense\*\*](#)

Checks whether you have license to the module.

#### [\*\*FRAppClearCustomRecentFileList\*\*](#)

Clears the recent files from the recent file list.

#### [\*\*FRAppClearRecentFileList\*\*](#)

Clears the recent opened file list.

#### [\*\*FRAppCollectBitmapData\*\*](#)

Notifies the other modules to collect some bitmap data.

#### [\*\*FRAppCollectDocActionData\*\*](#)

The plug-in can invoke this interface to notify the other module to collect the action data.

#### [\*\*FRAppCollectNormalData\*\*](#)

Notifies the other modules to collect some normal data.

#### [\*\*FRAppCollectNormalData2\*\*](#)

Notifies the other modules to collect some normal data.

#### [\*\*FRAppCommandLineGetArgument\*\*](#)

Gets the specified argument.

#### [\*\*FRAppCommandLineGetArgumentCount\*\*](#)

Gets the count of the argument.

**[FRAppCommandLineGetSafeArgument](#)**

Gets the specified argument. If the specified argument is not existing, the default value will be returned.

**[FRAppCommandLineHasSwitch](#)**

Checks whether the *pwszSwitch* exists.

**[FRAppConvertToPDF](#)**

convert file to Pdf file.

**[FRAppConvertToPDFA](#)**

Converts file to PDFA file.

**[FRAppCountAllTypesVisibleDocuments](#)**

Gets the count of the visible documents of all types, including the start page.

**[FRAppCountDocPropertyPages](#)**

Counts the doc property pages.

**[FRAppCountDocsOfPDDoc](#)**

Gets the number of document views. This method just only count the document view that for displaying PDF file type, does not count other document view for displaying other format file.

**[FRAppCountToolbars](#)**

Gets the number of toolbars, including the flyout toolbars.

**[FRAppCountTools](#)**

Gets the number of registered [FR\\_Tool](#) object.

**[FRAppCreateAnEmptyFrameWnd](#)**

Creates a new frame window.

**[FRAppCreateAnEmptyFrameWnd2](#)**

Creates a new frame window.

**[FRAppCreateBlankDoc](#)**

Creates a blank document.

**[FRAppCreateCustomRecentFileList](#)**

Creates a custom recent file list.

**[FRAppCreateEmptyFrameViewEventHandler](#)**

Creates the empty frame view event handler. Registers it through

[FRAppCreateAnEmptyFrameWnd2](#) .Destroys it through

[FRAppDestroyEmptyFrameViewEventHandler](#) ;

**[FRAppDestroyEmptyFrameViewEventHandler](#)**

Destroys the event handler of the empty frame window view

**[FRAppDestroyRibbonRecentFileEventHandler](#)**

Destroys the Ribbon recent file list event handler returned by

[FRAppRegisterRibbonRecentFileEventHandler](#) .

**[FRAppDestroySubscriptionProvider](#)**

Destroys the subscription provider returned by [FRAppSetSubscriptionProvider](#) .

**[FRAppDisableAllPanel](#)**

Disables all the navigation panels or not.

**[FRAppEndFullScreen](#)**

Ends full-screen mode. It does nothing if the the application is not running in full-screen mode.

**[FRAppExitOwnUndoMode](#)**

Exits the own undo mode.

**[FRAppExitReadMode](#)**

Exits the reading mode.

**[FRAppFZipUnzip](#)**

Unzips the .fzip file to the specified dest path.

**[FRAppGetActiveDocOfPDDoc](#)**

Gets the top-most document window which for displaying PDF file type.

**FRApGetActiveTool**

Gets the active tool from the application.

**FRApGetAdvWnd**

Gets the window handle to show the advertisement.

**FRApGetAppDataPath**

Gets the execution file path of current running reader.

**FRApGetAppName**

Gets the [FS\\_WideString](#) corresponding to the application's title.

**FRApGetCurMeasurementUnits**

Gets the current measurement unites.

**FRApGetCurrentChildFrm**

Gets the current child frame handle.

**FRApGetCustomRecentFileListItemCount**

Gets the item count of the custom recent file list.

**FRApGetDocOfPDDoc**

Gets the document view by specified index.

**FRApGetECMFileDialog**

Registers a security handler. Invoked this interface to process the PDF documents that are encrypted by customer security handler. Since: [SDK\\_LATEEST\\_VERSION](#) > 9.5

**FRApGetEditionType**

Gets the edition type. There are five types, Business, Standard, Express, Enterprise and Free.

**FRApGetGEModule**

Gets the app GE module.

**FRApGetKeyfileStartAndExpireDate**

Gets the start date and the expire date of the key file.

**FRApGetLocalFontName**

Gets the local font name.

**FRApGetMainframeShow**

Checks whether the main frame will be shown or not when Foxit Reader starts up.

**FRApGetMainFrameWnd**

Gets the main frame window of Foxit Reader.

**FRApGetMainFrameWndByIndex**

**FRApGetMainFrameWndCount**

**FRApGetMenuBar**

Gets Foxit Reader's menu bar.

**FRApGetModelessParentWnd**

**FRApGetModuleFileName**

Gets the module file name of application.

**FRApGetModuleMgr**

Gets the app module manager.

**FRApGetMousePos**

Gets the mouse position. The mouse position is specified in screen coordinates.

**FRApGetName**

Gets the [FS\\_BytString](#) corresponding to the application's name, which is the name of the file containing the Foxit Reader application.

**FRApGetNavPanelCount**

Gets the count of the navigation panel.

**FRApGetNavPanelNameByIndex**

Gets the name of the navigation panel.

**[FRAppGetNavPanelViewByName](#)**

Gets the window handle of the navigation panel view.

**[FRAppGetOEMVersion](#)**

Gets the OEM version name.

**[FRAppGetPageContextMenuPos](#)**

Gets the position of context menu on the page view.

**[FRAppGetPassphrase](#)**

This interface is invoked by the plug-in to confirm that the host environment is Foxit Reader and is legal. The following steps show the procedure:

- 1. Plug-in generates a random string of 128 byte length and encrypts it using the public key.
- 2. Plug-in allocates one buffer of 128 bytes and calls [FRAppGetPassPhrase](#) () and passes the encrypted string.
- 3. Foxit reader will decrypt the encrypted text using the private key and copy it to the buffer *pstrPlainText*.
- 4. Plug-in matches the plain text against what it had generated.

**[FRAppGetRecentFileIndex](#)**

Gets the index of the specified recent file.

**[FRAppGetRecentFileList](#)**

Gets the recent opened file list.

**[FRAppGetRecentFileListSize](#)**

Gets the recent file list size.

**[FRAppGetRecentFolderListSize](#)**

Gets the recent folder list size.

**[FRAppGetRibbonBar](#)**

Gets Foxit Reader's ribbon bar.

**[FRAppGetRibbonBar2](#)**

Gets Foxit Reader's ribbon bar.

**[FRAppGetRibbonBarBackGroundColor](#)**

Gets the back ground color of ribbon bar.

**[FRAppGetRibbonBarBtnBackGroundColor](#)**

Gets the back ground color of ribbon bar button.

**[FRAppGetRibbonBaseBorderColor](#)**

Gets the back ground color of ribbon base border.

**[FRAppGetRibbonCategoryColor](#)**

Gets the back ground color of ribbon category.

**[FRAppGetRibbonElementColor](#)**

Gets the back ground color of ribbon element.

**[FRAppGetRibbonTooltipBorderColor](#)**

Gets the border color of the ribbon tooltip.

**[FRAppGetStartAppMode](#)**

Gets the app starting mode, 0 for not open any documents, and 1 for opening some documents at the same time.

**[FRAppGetStartMenuOfTabbedToolbarMode](#)**

Gets the start menu which is in tabbed toolbar mode.

**[FRAppGetStatusBarBkGroundColor](#)**

Gets the back ground color of status bar in ribbon mode.

**[FRAppGetStatusBarBkGroundPath](#)**

Gets the path of the back ground color picture of the status bar in classic mode.

**[FRAppGetTaskPaneWndHandleByDoc](#)**

Gets the window handle of the task pane.

**FRAppGetToolBar**

Gets the specified toolbar.

**FRAppGetToolBarByName**

Gets toolbar created with the specified name, All toolbars(including flyout toolbar) can be obtained by the method.

**FRAppGetToolbarLocked**

Checks whether the toolbar is locked.

**FRAppGetToolByIndex**

Gets the specified tool.

**FRAppGetToolByName**

Gets the FR\_Tool object that was registered under specified name.

**FRAppGetUIParentWndByTaskPane**

Gets the UI parent window handle.

**FRAppGetVersion**

Gets the FS\_WideString corresponding to the application's version. The Format of version information is XX.XX.XX.XX(major number.minor number.maintainence number.build number).

**FRAppIsCurDocShowInBrowser**

Tests whether the current document is opened in browser or not.

**FRAppIsDisableAllPanel**

Checks whether all the panels are disabled.

**FRAppIsFipsMode**

Whether the system is in Federal Information Processing Standard environment or not.

**FRAppIsFullScreen**

Tests whether the application is running in full-screen mode.

**FRAppIsJSEnabled**

Checks whether the javascript is enabled or not.

**FRAppIsLicenseValidOrInTrial**

Checks whether the license is valid or is in trial.

**FRAppIsMainfrmActivating**

**FRAppIsNeedCollectData**

Whether the cPDF data is need collected or not.

**FRAppIsPDF2Doc**

whether it is 2.0 pdf document

**FRAppIsReaderOnlyMode**

**FRAppIsReadMode**

Checks whether Foxit Reader is in reading mode or not.

**FRAppIsRibbonMode**

Tests whether current toolbar mode is tabbed mode.

**FRAppIsRTLLanguage**

Checks whether the current language of app is right-to-left or not.

**FRAppIsRunEmbedded**

Tests whether the Foxit Reader is running embedded or not.

**FRAppLoadLibrary**

Load the library.

**FRAppModalWindowIsOpen**

A client should use this method to determine whether a modal window is open. There is a large (and ill-defined) group of actions that are illegal while a modal window is open, although these actions are not programmatically prevented by the Foxit Reader Viewer.

While a modal dialog box is open, a client must not open documents, change pages, change

views, close documents, change tools, or do anything that might disrupt the user or Foxit Reader viewer.

**FRAppOnCmdMsgByName**

Routes and dispatches command messages from plug-ins to plug-ins

**FRAppOpenFileAttachment**

open a embed file.

**FRAppPopDocPropertyPage**

Pops up the doc property pages.

**FRAppRedrawRecentFileList**

Redraws the recent file list in Ribbon mode.

**FRAppRefreshLayerPanelView**

**FRAppRegisterActionHandler**

Registers an action handler. The callbacks are called by Foxit Reader when the Foxit Reader will do the actions. You can implement your own process to customize the action process.

**FRAppRegisterAppEventHandler**

Registers a user-supplied procedure set to call when some application level event occurs.

**FRAppRegisterCaptureHandler**

Registers the capture handler.

**FRAppRegisterCmdMsgEventHandler**

Registers a callbacks called by the Foxit Reader to route and dispatch command messages and to handle the update of command user-interface objects, such as menu, toolbar.

**FRAppRegisterContentProvider**

Registers a content provider so that the plug-in can process the protected document and provide decrypted document data.

**FRAppRegisterDataCollectionHadler**

Registers the data collection handler to collect the data as you need.

**FRAppRegisterDocHandlerOfPDDoc**

Registers a user-supplied event handler to document window.

**FRAppRegisterDocPropertyPageHandler**

Registers the event notification to document properties dialog.

**FRAppRegisterExtraPrintInfoProvider**

Register the printing notifies.

**FRAppRegisterForContextMenuAddition**

Registers a user-supplied procedure to call after a context menu has been created but before it is shown to the user. The callback can add menu items to or remove menu items from the menu. \*\*\*\*\*

**FRAppRegisterMousePtHandler**

Registers the mouse point handler.

**FRAppRegisterNavPanelView**

Registers a navigation panel view into the Foxit Reader.

**FRAppRegisterOwnerFileType**

Adds a file type to Foxit Reader. You can control the process of owner file type, such as opening, saving, sending email and so on.

**FRAppRegisterPageHandlerOfPDDoc**

Registers a page-level event callback set.

**FRAppRegisterPDFAPPluginHandler**

The callbacks of pdfa Handler just be called while the document was saved to pdfa.

**FRAppRegisterPOEventHandler**

Registers a callbacks called by the Foxit Reader to do the page organizing, such as adding pages, deleting pages, replacing pages and so on.

**FRAppRegisterPreferencePageHandler**

Registers the event notification to preference dialog.

**FRAppRegisterRibbonRecentFileEventHandler**

Registers a callbacks set for Ribbon recent file list event handler.

**FRAppRegisterSecurityHandler**

Registers a security handler. Invoked this interface to process the PDF documents that are encrypted by customer security handler.

**FRAppRegisterSecurityMethod**

Registers the security method that you can manage your security method.

**FRAppRegisterSelectionHandler**

Registers a new selection handler. Selection handlers allow the selection of items other than those that can be selected in the as-shipped Foxit viewer. For example, a selection handler could allow a user to select a sampled image. This method can be used to replace an existing selection handler that handles the same selection type.

**FRAppRegisterTaskPaneView**

Registers callback set for a task pane view.

**FRAppRegisterTool**

Registers a tool into the Foxit Reader.

**FRAppRegisterWinMSGHandler**

Registers callback set for a windows MSG handler.

**FRAppRegisterWndProvider**

Registers a window provider to create a window above the document view. There are already some windows above the document view, such as PDF view, text view, ruler view.

**FRAppRegistryGetBinary**

Gets the binary value in the specified section.

**FRAppRegistryGetEntryCount**

Gets the entry count in the specified section.

**FRAppRegistryGetEntryName**

Gets the entry name in the specified section.

**FRAppRegistryGetInt**

Gets the int value in the specified section.

**FRAppRegistryGetKeyCounts**

Gets the counts of key for the specified section.

**FRAppRegistryGetKeyName**

Gets the key name in the specified section.

**FRAppRegistryGetProfilePath**

Gets the registry's profile path of the Foxit Reader.

**FRAppRegistryGetString**

Gets the string value in the specified section.

**FRAppRegistryRemoveEntry**

Removes the specified entry.

**FRAppRegistryRemoveSection**

Removes the specified section.

**FRAppRegistryWriteBinary**

Write binary value to the specified registry. If you want to set the value in registry like "HKEY\_CURRENT\_USER\Software\Foxit Software\Foxit Reader\plugins\Create Link", lpszSection can be set as "Create Link". If you want to set the value in registry like "HKEY\_CURRENT\_USER\Software\Foxit Software\Foxit Reader\plugins\Create Link\Annot\" lpszSection can be set as "Create Link\Annot".

**FRAppRegistryWriteInt**

Write int value to the specified registry. If you want to set the value in registry like "HKEY\_CURRENT\_USER\Software\Foxit Software\Foxit Reader\plugins\Create Link", lpszSection can be set as "Create Link". If you want to set the value in registry like

"HKEY\_CURRENT\_USER\Software\Foxit Software\Foxit Reader\plugins\Create Link\Annot\" lpszSection can be set as "Create Link\Annot".

**FRAppRegistryWriteString**

Write string to the specified registry. If you want to set the value in registry like "HKEY\_CURRENT\_USER\Software\Foxit Software\Foxit Reader\plugins\Create Link", lpszSection can be set as "Create Link". If you want to set the value in registry like "HKEY\_CURRENT\_USER\Software\Foxit Software\Foxit Reader\plugins\Create Link\Annot\" lpszSection can be set as "Create Link\Annot".

**FRAppReleaseNavPanelView**

Releases the navigation panel view returned by [FRAppAddNavPanelView](#).

**FRAppReloadStartPage**

Reloads the start page.

**FRAppRemoveFileFromCustomRecentFileList**

Removes the recent file path from the specified recent file list.

**FRAppRemoveFileFromRecentFileList**

Removes the recent opened file path from the list.

**FRAppRemoveTaskPanelByName**

Removes the specified task panel.

**FRAppRestartProcess**

Restarts the process of Foxit Reader or Foxit PhantomPDF.

**FRAppSetActiveDocOfPDDoc**

Sets a document view to top-most.

**FRAppSetActiveTool**

Sets the active tool. It does nothing if the specified tool is not currently enabled, The [FRToolIsEnabled\(\)](#) callback in [FR\\_ToolCallbacksRec](#) structure determines whether a tool is enabled. If this callback is [NULL](#), the tool is always enabled.

**FRAppSetCustomRecentFileListItemBitmap**

Sets the icon that will be displayed in front of the recent file list label.

**FRAppSetCustomRecentFileListItemTitle**

Sets the title of the recent file list item.

**FRAppSetCustomRecentFileListItemTooltip**

Sets the tooltip of the recent file.

**FRAppSetCustomRecentFileListLabel**

Sets the label of the custom recent file list.

**FRAppSetCustomRecentFileListMaxSize**

Sets the max size of recent file list item.

**FRAppSetEnableJS**

Sets the javascript to be enabled or not.

**FRAppSetMainframeShow**

Sets whether the main frame will be shown or not when Foxit Reader starts up.

**FRAppSetMetadataOption**

Sets the option to the document to control whether the metadata is to be compressed or not.

**FRAppSetOwnUndoMode**

Sets the own undo mode.

**FRAppSetRecentFileListImageByExt**

Sets the image of the file in the recent file list.

**FRAppSetSubscriptionProvider**

Registers a callbacks set for subscription provider.

**FRAppSetToolbarLocked**

Sets the toolbar to be locked or not.

**FRAppShowAdvWnd**

Shows the advertisement window or not.

**FRAppShowFullScreen**

Begins full-screen mode. In full-screen mode, all window borders, the menu bar, and the toolbar are hidden. All regions of the screen outside of the page view boundary are painted by specified color. [FRAppShowFullScreen](#) () is ignored if the application is already in full-screen mode, or if there are no currently open documents.

**FRAppShowMenuBar**

Shows/Hides menu bar.

**FRAppShowPreferenceDlg**

Shows the preference dialog.

**FRAppShowTaskPane**

Shows or hidden a task panel view.

**FRAppUndoRedoBeginEdit**

When you begin to edit detail content, such as text, and you don't want to save all the editing operation to save memory, you can invoke this interface. We call it the editing mode undo-redo control.

**FRAppUndoRedoEndEdit**

When you exit the editing mode, invoke this interface to exit the editing mode undo-redo control.

**FRAppUndoRedoIsEditing**

Checks whether the document is being edited.

**FRAppUnRegisterCaptureHandler**

Unregisters the capture handler and releases the memory.

**FRAppUnRegisterCmdMsgEventHandler**

Unregisters the input cmd msg event.

**FRAppUnRegisterMousePtHandler**

Unregisters the mouse point handler and releases the memory.

**FRAppUnRegisterSecurityHandler**

Unregisters a security handler.

**FRAppUnRegisterSelectionHandler**

Unregisters the selection handler and releases the memory.

**FRAppUnRegisterWndProvider**

Unregisters the window provider by name.

## Functions detail

**FRAppActiveChildFrame****Syntax**

```
void FRAppActiveChildFrame (
    HWND hChildFrame
);
```

**Description**

Activates the specified child frame.

**Parameter**

---

hChildFrame	[In] The input child frame handle to be activated.
-------------	--

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1880

**Related method****Since**[SDK LATEEST VERSION > 7.3.1](#)**FRAppAddDocPropertyPage****Syntax**

```
FS_BOOL FRAppAddDocPropertyPage (
    HWND parent,
    HWND propertyPage,
    FS_LPCWSTR title
);
```

**Description**

Adds a document property page into Reader document properties dialog.

**Parameter**

parent	[In] The parent window.
propertyPage	[In] A platform-specific object. In windows, it is a <i>HWND</i> handler.
title	[In] The title of the document property page. It will be displayed on the tabsheet in the top of document properties dialog.

**Return**

[TRUE](#) for success, or [FALSE](#) if the property page has existed or the title of the page has been used.

**Head file reference**

fr\_appTempl.h: 290

**Related method**[FRAppRegisterDocPropertyPageHandler](#)

**Note:** You should call this method in the implementation of `FRDocPropertyPageOnCreate()` callback.

**FRAppAddFileAttachment**

**Syntax**

```
FS_BOOL FRAppAddFileAttachment (
    FR\_Document frDoc
);
```

**Description**

Add a embed file to current document with The file path.

**Parameter**


---

frDoc	[In] The specified document. param[in]: lpwsAttachFilePath The file path for add.
-------	--

---

**Return**

Non-zero means succeed, otherwise failure.

**Head file reference**

fr\_appTempl.h: 2048

**Since**

[SDK\\_LATEEST\\_VERSION > 9.1](#)

**FRAppAddFileToCustomRecentFileList****Syntax**

```
FS_BOOL FRAppAddFileToCustomRecentFileList (
    FS\_LPCSTR lpsRecentFileName,
    FS LPCWSTR lpwsFilePath,
    FS LPCWSTR lpwsFileTitle,
    FS LPCWSTR lpwsFileTooltip,
    FRAppCustomRecentFileExecuteProc exeProc,
    void* clientData
);
```

**Description**

Adds the recent file to the custom recent file list.

**Parameter**


---

lpsRecentFileName	[In] The name that specifies the custom recent file list which the recent file path will be added to.
-------------------	---

---

lpwsFilePath	[In] The recent file path that will be added.
--------------	---

---

lpwsFileTitle	[In] Instead of the file path, the title will be displayed in the recent file list.
---------------	---

---

---

lpwsFileTooltip	[In] The tooltip of the recent file.
exeProc	[In] When the recent file item is clicked, the callback will be invoked, unless you set it as NULL.
clientData	[In] The client data that will be passed to the callback.

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1056

**Related method**

[FRAppCreateCustomRecentFileList](#)

**Since**

[SDK LATEST VERSION > 1.0](#)

## FRAppAddFileToCustomRecentFileList2

**Syntax**

```
FS_BOOL FRAppAddFileToCustomRecentFileList2 (
    FS_LPCSTR lpsRecentFileName,
    FS_LPCWSTR lpwsFilePath,
    FS_LPCWSTR lpwsFileTitle,
    FS_LPCWSTR lpwsFileTooltip,
    FRAppCustomRecentFileExecuteProc exeProc,
    FS_BOOL bEnableRemoveItem,
    FS_INT32 nPinned,
    void* clientData
);
```

**Description**

Adds the recent file to the custom recent file list. Registers the Ribbon recent file event handler to catch the pinning and removing event, so that the plug-in can update the data layer.

**Parameter**


---

lpsRecentFileName	[In] The name that specifies the custom recent file list which the recent file path will be added to.
lpwsFilePath	[In] The recent file path that will be added.
lpwsFileTitle	[In] Instead of the file path, the title will be displayed in the recent file list.

---

---

lpwsFileTooltip	[In] The tooltip of the recent file.
exeProc	[In] When the recent file item is clicked, the callback will be invoked, unless you set it as NULL.
bEnableRemoveItem	[In] Whether the item can be removed or not.
nPinned	[In] The status of the pin, 0 for not pinned, 1 for pinned, 2 for not display the pin button.
clientData	[In] The client data that will be passed to the callback.

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1666

**Related method**

[FRAppCreateCustomRecentFileList](#)

[FRAppRegisterRibbonRecentFileEventHandler](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRAppAddFileToRecentFileList****Syntax**

```
void FRAppAddFileToRecentFileList (
    FS\_LPCWSTR lpszPathName
);
```

**Description**

Adds the recent opened file path to the list.

**Parameter**


---

lpszPathName	[In] The recent opened file path.
--------------	-----------------------------------

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 870

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FRAppAddLog****Syntax**

```
void FRAppAddLog (
    FR_LOG_LEVEL nLevel,
    FS_WideString wsMsg,
    const FS_CHAR* file,
    FS_INT32 nLine,
    const FS_CHAR* function
);
```

**Description**

Outputs log information.

**Parameter**

nLevel	[In] The specified log level.
wsMsg	[In] The input log message.
file	[In] The file where the log is added. Use __FILE__ as default.
nLine	[In] The code line where the log is added. Use __LINE__ as default.
function	[In] The function where the log is added. use __FUNCTION__ as default.

**Return**

void.

**Head file reference**

fr\_appTempl.h: 2120

**Since**[SDK\\_LATEEST\\_VERSION > 9.1](#)**FRAppAddNavPanelView****Syntax**

```
void* FRAppAddNavPanelView (
    FR_PanelViewCallbacks panelViewCallbacks,
    FS_INT32 nPos
```

);

**Description**

Adds a navigation panel view into the Foxit Reader. The navigation panel view can be added after reader has started up.

**Parameter**

---

panelViewCallbacks	[In] A panel view callback set invoked by reader to add a panel view to navigation panel.
--------------------	---

---

nPos

[In] The position in the navigation panel.

---

**Return**

The returned value should be released by [FRAppReleaseNavPanelView](#) .

**Head file reference**

fr\_appTempl.h: 2027

**Since**

[SDK LATEEST VERSION > 8.3.2](#)

**FRAppAddPreferencePage****Syntax**

```
FS_BOOL FRAppAddPreferencePage (
    HWND parent,
    HWND preferPage,
    FS_LPCWSTR title
);
```

**Description**

Adds a preference page into Reader preference dialog.

**Parameter**

---

parent	[In] The parent window.
--------	-------------------------

---

preferPage

[In] A platform-specific object. In windows, it is a HWND handler.

---

title

[In] The title of the preference page. It will be displayed on the list box in the left of preference dialog.

---

**Return**

---

[TRUE](#) for success, or [FALSE](#) if the preference page has existed or the title of the page has been used.

**Head file reference**

fr\_appTempl.h: 262

**Related method**[FRAppRegisterPreferencePageHandler](#)

**Note:** You should call this method in the implementation of FRPrefPageOnCreate() callback.

**FRAppAddUndoRedoItem****Syntax**

```
void FRAppAddUndoRedoItem (
    FS\_LPCWSTR lpwDescr,
    FR\_Document frDoc,
    FS\_BOOL bEdit,
    FR\_UndoRedoCallbacks callbacks
);
```

**Description**

Adds a undo-redo item to the specified document.

**Parameter**


---

lpwDescr	[In] The input description of the undo-redo item.
frDoc	[In] The specified document.
bEdit	[In] Whether the current operation is editing, for example, you add a typewriter and enter characters.
callbacks	[In] The user-supplied callbacks for implementing undo-redo.

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 498

**Related method**

[FRAppUndoRedoIsEditing](#)  
[FRAppUndoRedoBeginEdit](#)  
[FRAppUndoRedoEndEdit](#)

## FRApAdvWndSetCustomize

### Syntax

```
void FRApAdvWndSetCustomize (
    FS_BOOL bCustomize
);
```

### Description

Sets whether to customize the advertisement.

### Parameter

---

bCustomize	[In] Whether to customize the advertisement.
------------	--

---

### Return

void.

### Head file reference

fr\_appTempl.h: 1484

### Related method

#### Since

[SDK LATEEST VERSION > 2.1.0.4](#)

## FRApCanQuit

### Syntax

```
FS_BOOL FRApCanQuit (void );
```

### Description

Gets the permission to decide whether you can quit.

### Return

[TRUE](#) if foxit reader can quit, [FALSE](#) if cannot. The default version of the routine always returns [TRUE](#).

### Head file reference

fr\_appTempl.h: 127

## FRApCheckModuleLicense

### Syntax

```
FS_BOOL FRApCheckModuleLicense (
    FS_LPCWSTR lpwszModuleName
);
```

**Description**

Checks whether you have license to the module.

**Parameter**

---

lpwszModuleName	[In] The input module name.
-----------------	-----------------------------

---

**Return**

Returns [TRUE](#) if you have license to the module.

**Head file reference**

fr\_appTempl.h: 921

**Note:** This is an internal interface.

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRApClearCustomRecentFileList****Syntax**

```
void FRAppClearCustomRecentFileList (
    csRecentListName
);
```

**Description**

Clears the recent files from the recent file list.

**Parameter**

---

csRecentListName	[In] The name that specifies the custom recent file list whose recent files will be cleared.
------------------	--

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 1072

**Related method**

[FRAppCreateCustomRecentFileList](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRAppClearRecentFileList****Syntax**

```
void FRAppClearRecentFileList (void );
```

**Description**

Clears the recent opened file list.

**Return**

void.

**Head file reference**

fr\_appTempl.h: 880

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRAppCollectBitmapData****Syntax**

```
void FRAppCollectBitmapData (
    FS_LPCWSTR lpwsFunction,
    FS_LPCWSTR lpwsAction,
    FS_DIBitmap pBitmap,
    FS_INT32 nLevel
);
```

**Description**

Notifies the other modules to collect some bitmap data.

**Parameter**

lpwsFunction	[In] The function to be collected.
lpwsAction	[In] The function to be collected.
pBitmap	[In] The bitmap to be collected.
nLevel	[In] The detail level of the data.

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1993

**Related method**[FRAppRegisterDataCollectionHadler](#)**Since**[SDK\\_LATEEST\\_VERSION > 8.2.1](#)**FRAppCollectDocActionData****Syntax**

```
void FRAppCollectDocActionData (
    FR\_Document frDoc,
    FS\_LPCWSTR lpwsOperatorType,
    FS\_LPCWSTR lpwsOperator,
    FS\_MapPtrToPtr valueMap
);
```

**Description**

The plug-in can invoke this interface to notify the other module to collect the action data.

**Parameter**


---

<a href="#">frDoc</a>	[In] The specified <a href="#">FR_Document</a> .
-----------------------	--

---

<a href="#">lpwsOperatorType</a>	[In] The specified operator type.
----------------------------------	-----------------------------------

---

<a href="#">lpwsOperator</a>	[In] The operator value.
------------------------------	--------------------------

---

<a href="#">valueMap</a>	[In] The operator value map.
--------------------------	------------------------------

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1866

**Related method**[FRDocOnDocCollectActionData](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.3.1](#)**FRAppCollectNormalData****Syntax**

```
void FRAppCollectNormalData (
    FS\_LPCWSTR lpwsFunction,
```

---

```
FS LPCWSTR lpwsAction,
FS LPCWSTR lpwsContent
);
```

**Description**

Notifies the other modules to collect some normal data.

**Parameter**


---

lpwsFunction	[In] The function to be collected.
lpwsAction	[In] The function to be collected.
lpwsContent	[In] The content to be collected.

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1468

**Related method**

[FRAppRegisterDataCollectionHadler](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRAppCollectNormalData2****Syntax**

```
void FRAppCollectNormalData2 (
    FS LPCWSTR lpwsFunction,
    FS LPCWSTR lpwsAction,
    FS LPCWSTR lpwsContent,
    FS INT32 nLevel
);
```

**Description**

Notifies the other modules to collect some normal data.

**Parameter**


---

lpwsFunction	[In] The function to be collected.
lpwsAction	[In] The function to be collected.

---

---

lpwsContent	[In] The content to be collected.
-------------	-----------------------------------

---

nLevel	[In] The detail level of the data.
--------	------------------------------------

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1979

**Related method**[FRAppRegisterDataCollectionHadler](#)**Since**[SDK\\_LATEEST\\_VERSION > 8.2.1](#)**FRAppCommandLineGetArgument****Syntax**

```
void FRAppCommandLineGetArgument (
    FS_LPCWSTR pwszSwitch,
    FS_INT32 iIndex,
    FS_WideString* outArgument
);
```

**Description**

Gets the specified argument.

**Parameter**


---

pwszSwitch	[In] The switch name to which the argument belongs.
------------	---

---

iIndex	[In] The index of the arguments.
--------	----------------------------------

---

outArgument	[Out] It receives the argument.
-------------	---------------------------------

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 828

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRAppCommandLineGetArgumentCount****Syntax**

```
FS_INT32 FRAppCommandLineGetArgumentCount (
    FS_LPCWSTR pwszSwitch
);
```

**Description**

Gets the count of the argument.

**Parameter**


---

pwszSwitch	[In] The switch name to which the argument belongs.
------------	---

---

**Return**

The count of the argument.

**Head file reference**

fr\_appTempl.h: 840

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRAppCommandLineGetSafeArgument****Syntax**

```
void FRAppCommandLineGetSafeArgument (
    FS_LPCWSTR pwszSwitch,
    FS_INT32 iIndex,
    FS_LPCWSTR pwszDefault,
    FS_WideString* outArgument
);
```

**Description**

Gets the specified argument. If the specified argument is not existing, the default value will be returned.

**Parameter**


---

pwszSwitch	[In] The switch name to which the argument belongs.
------------	---

---

iIndex	[In] The index of the arguments.
--------	----------------------------------

---

pwszDefault	[In] The default value of argument. If the specified argument is not existing, the default value will be returned.
-------------	--

---

---

outArgument	[Out] It receives the argument.
-------------	---------------------------------

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 815

**Since**[SDK LATEST VERSION > 1.0](#)**FRAppCommandLineHasSwitch****Syntax**

```
FS_BOOL FRAppCommandLineHasSwitch (
    FS_LPCWSTR pwszSwitch
);
```

**Description**Checks whether the *pwszSwitch* exists.**Parameter**


---

pwszSwitch	[In] The switch name to be checked.
------------	-------------------------------------

---

**Return**[TRUE](#) if the switch exists, otherwise [FALSE](#)**Head file reference**

fr\_appTempl.h: 804

**Related method**[FRAppOnRunCommandLine](#)**Since**[SDK LATEST VERSION > 1.0](#)**FRAppConvertToPDF****Syntax**

```
FS_INT32 FRAppConvertToPDF (
    lpwsFile,
    FS_LPCWSTR lpwsDest,
    HWND hWnd,
    FS_BOOL bShowProgressBar,
    FS_BOOL bConvertToCPDF,
    FS_BOOL bDelDestFile
);
```

**Description**

convert file to Pdf file.

**Parameter**


---

lpwsFile	[In] The Original File.
lpwsDest	[In] The Dest File.
hWnd	[In] The hWnd, default NULL.
bShowProgressBar	[In] If show the progress bar.
bConvertToCPDF	[In] If need convert to CPDF File.
bDelDestFile	[In] If need delete the Dest File when convert failed.

---

**Return**

zero means successful, otherwise failed.

**Head file reference**

fr\_appTempl.h: 2081

**Related method**

[FRAppConvertToPDFA](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 9.1](#)

**FRAppConvertToPDFA****Syntax**

```
FS_INT32 FRAppConvertToPDFA (
    FR\_Document frDoc,
    const FS\_WideString fswsSaveAsFileName,
    const FRPDFA\_PDFVersion pVersion
);
```

**Description**

Converts file to PDFA file.

**Parameter**

---

frDoc	[In] The specified document.
fswsSaveAsFileName	[In] The dest document path to be saved.
pVersion	[In] The PDFA version to be converted.

---

**Return**

Zero means successful, otherwise failed.

**Head file reference**

fr\_appTempl.h: 2092

**Related method**

[FRAppConvertToPDF](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 9.1](#)

**FRAppCountAllTypesVisibleDocuments****Syntax**

```
FS_INT32 FRAppCountAllTypesVisibleDocuments (void );
```

**Description**

Gets the count of the visible documents of all types, including the start page.

**Return**

The count of the visible documents of all types, including the start page.

**Head file reference**

fr\_appTempl.h: 1657

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

**FRAppCountDocPropertyPages****Syntax**

```
FS_INT32 FRAppCountDocPropertyPages (
    HWND hParent
);
```

**Description**

Counts the doc property pages.

**Parameter**

---

<b>hParent</b>	[In] The parent window.
----------------	-------------------------

---

**Return**

The count of doc property pages.

**Head file reference**

fr\_appTempl.h: 910

**Related method**

[FRAppRegisterDocPropertyPageHandler](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRAppCountDocsOfPDDoc****Syntax**

```
FS_INT32 FRAppCountDocsOfPDDoc (void );
```

**Description**

Gets the number of document views. This method just only count the document view that for displaying PDF file type, does not count other document view for displaying other format file.

**Return**

The number of document views that is used to display PDF file type.

**Head file reference**

fr\_appTempl.h: 330

**Related method**

[FRAppGetDocOfPDDoc](#)

**FRAppCountToolbars****Syntax**

```
FS_INT32 FRAppCountToolbars (void );
```

**Description**

Gets the number of toolbars, including the flyout toolbars.

**Return**

The number of toolbar.

**Head file reference**

fr\_appTempl.h: 136

**Related method**[FRAppGetToolBar](#)**FRAppCountTools****Syntax**

```
FS_INT32 FRAppCountTools (void );
```

**Description**

Gets the number of registered [FR\\_Tool](#) object.

**Return**

The number of tools.(Including built-in tools)

**Head file reference**

fr\_appTempl.h: 209

**Related method**[FRAppGetToolByIndex](#)**FRAppCreateAnEmptyFrameWnd****Syntax**

```
HWND FRAppCreateAnEmptyFrameWnd (
    FS_WideString title,
    FS_BOOL bMakeVisible,
    FR_EmptyFramWndNotifies notify
);
```

**Description**

Creates a new frame window.

**Parameter**

---

title	[In] The title of the frame which will be created.
-------	--

---

bMakeVisible	[In] A flag indicate whether to show the frame.
--------------	---

---

notify	[In] The notifies for new window.
--------	-----------------------------------

---

**Return**

The handler of the newly created frame.

**Head file reference**

fr\_appTempl.h: 722

**FRAppCreateAnEmptyFrameWnd2****Syntax**

```
HWND FRAppCreateAnEmptyFrameWnd2 (
    FS_LPCWSTR lpwsTitle,
    FS_LPCWSTR lpwsPathName,
    FR_EmptyFrameWndViewEventHandler eventHandler,
    FS_BOOL bCreatePanel,
    FS_BOOL bMakeVisible,
    FS_BOOL bAddToMRU
);
```

**Description**

Creates a new frame window.

**Parameter**

lpwsTitle	[In] The title of the frame which will be created.
lpwsPathName	[In] The input document path associated with the view.
eventHandler	[In] The input empty frame window view event handler.
bCreatePanel	[In] Whether to create the associated navigation panel or not. Sets it FALSE as default.
bMakeVisible	[In] A flag indicate whether to show the frame.
bAddToMRU	[In] Determines whether the file name is added to the most recently used (MRU) file list. Sets it FALSE as default.

**Return**

The handler of the newly created frame.

**Head file reference**

fr\_appTempl.h: 1796

**Related method**

[FRAppCreateEmptyFrameViewEventHandler](#)

[FRAppDestroyEmptyFrameViewEventHandler](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FRAppCreateBlankDoc**

**Syntax**

```
FR_Document FRAppCreateBlankDoc (
    FS_FLOAT fWidth,
    FS_FLOAT fHeight
);
```

**Description**

Creates a blank document.

**Parameter**


---

fWidth	[In] The width of the page in the blank document.
fHeight	[In] The height of the page in the blank document.

---

**Return**

The blank document.

**Head file reference**

fr\_appTempl.h: 1527

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 3.0](#)

**FRAppCreateCustomRecentFileList****Syntax**

```
FS_BOOL FRAppCreateCustomRecentFileList (
    FS_LPCSTR lpsRecentFileName,
    FS_LPCWSTR lpwsFileExt,
    FS_DIBitmap pItemBitmap,
    FS_LPCWSTR lpwsLabel,
    FS_INT32 nMaxSize
);
```

**Description**

Creates a custom recent file list.

**Parameter**


---

lpsRecentFileName	[In] The input custom recent file list name.
lpwsFileExt	[In] The extension of files that will be added to the recent file list.

---

---

<b>pItemBitmap</b>	[In] The icon that will be displayed in front of the recent file list label.
--------------------	--

---

<b>lpwsLabel</b>	[In] The input label of recent file list.
------------------	---

---

<b>nMaxSize</b>	[In] The input max size of recent file list item.
-----------------	---

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1030

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRAppCreateEmptyFrameViewEventHandler****Syntax**

```
FR_EmptyFrameWndViewEventHandler  
FRAppCreateEmptyFrameViewEventHandler ( FR\_EmptyFrameWndViewEventHandlerCallbacks callbacks  
);
```

**Description**

Creates the empty frame view event handler. Registers it through [FRAppCreateAnEmptyFrameWnd2](#). Destroys it through [FRAppDestroyEmptyFrameViewEventHandler](#);

**Parameter**


---

<b>callbacks</b>	[In] The callbacks for the event handler of the empty frame window view.
------------------	--

---

**Return**

The [FR\\_EmptyFrameWndViewEventHandler](#) object.

**Head file reference**

fr\_appTempl.h: 1795

**Related method**

[FRAppCreateAnEmptyFrameWnd2](#)

[FRAppDestroyEmptyFrameViewEventHandler](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FRAppDestroyEmptyFrameViewEventHandler****Syntax**

```
void FRAppDestroyEmptyFrameViewEventHandler (
    FR\_EmptyFrameWndViewEventHandler eventHandler
);
```

**Description**

Destroys the event handler of the empty frame window view

**Parameter**


---

eventHandler	[In] The input event handler of the empty frame window view.
--------------	--

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1796

**Related method**

[FRAppCreateAnEmptyFrameWnd2](#)

[FRAppCreateEmptyFrameViewEventHandler](#)

**Since**

[SDK LATEEST VERSION > 7.3](#)

**FRAppDestroyRibbonRecentFileEventHandler****Syntax**

```
void FRAppDestroyRibbonRecentFileEventHandler (
    void* eventHandler
);
```

**Description**

Destroys the Ribbon recent file list event handler returned by [FRAppRegisterRibbonRecentFileEventHandler](#).

**Parameter**


---

eventHandler	[In] The pointer to Ribbon recent file list event handler returned by <a href="#">FRAppRegisterRibbonRecentFileEventHandler</a> .
--------------	---

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1689

**Related method**

[FRAppRegisterRibbonRecentFileEventHandler](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRAppDestroySubscriptionProvider**

**Syntax**

```
void FRAppDestroySubscriptionProvider (
    void* subscriptionProvider
);
```

**Description**

Destroys the subscription provider returned by [FRAppSetSubscriptionProvider](#).

**Parameter**

---

subscriptionProvider	[In] The pointer to subscription provider returned by <a href="#">FRAppSetSubscriptionProvider</a> .
----------------------	--

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1722

**Related method**

[FRAppSetSubscriptionProvider](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRAppDisableAllPanel**

**Syntax**

```
void FRAppDisableAllPanel (
    FS_BOOL bDisable,
    HWND hFrameWnd
);
```

**Description**

Disables all the navigation panels or not.

**Parameter**

---

bDisable	[In] <b>TRUE</b> means all the navigation panels will be disabled, otherwise not.
hFrameWnd	[In] The child frame window. Set it as NULL to use the current child frame window.

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 1271

**Since**[SDK LATEEST VERSION > 2.0](#)**FRAppEndFullScreen****Syntax**

FS\_BOOL FRAppEndFullScreen (void );

**Description**

Ends full-screen mode. It does nothing if the application is not running in full-screen mode.

**Return**

**TRUE** if the application exit full-screen mode, otherwise not.

**Head file reference**

fr\_appTempl.h: 392

**Related method**[FRAppShowFullScreen](#)**FRAppExitOwnUndoMode****Syntax**

```
void FRAppExitOwnUndoMode (
    FR\_Document frDoc
);
```

**Description**

Exits the own undo mode.

**Parameter**


---

frDoc	[In] The input document.
-------	--------------------------

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 1333

**Related method**

[FRAppAddUndoRedoItem](#)  
[FRAppUndoRedoIsEditing](#)  
[FRAppUndoRedoBeginEdit](#)  
[FRAppUndoRedoEndEdit](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRAppExitReadMode****Syntax**

```
void FRAppExitReadMode (void );
```

**Description**

Exits the reading mode.

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1841

**Related method**

[FRAppIsReadMode](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRAppFZipUnzip****Syntax**

```
FS_BOOL FRAppFZipUnzip (  
    FS_LPCWSTR lpszFZipPath,  
    FS_LPCWSTR lpszDestPath  
) ;
```

**Description**

Unzips the .fzip file to the specified dest path.

**Parameter**

---

lpszFZipPath	[In] The input .fzip file path.
--------------	---------------------------------

---

lpszDestPath	[In] The dest path.
--------------	---------------------

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1949

**Since**

[SDK\\_LATEEST\\_VERSION > 8.1](#)

**FRAppGetActiveDocOfPDDoc****Syntax**

```
FR_Document FRAppGetActiveDocOfPDDoc (void );
```

**Description**

Gets the top-most document window which for displaying PDF file type.

**Return**

The top-most document window which is used for displaying PDF file type, or [NULL](#) if no documents are open. [NULL](#) is also returned while the top-most document window is not displaying PDF file type and while a document is being opened.

**Head file reference**

fr\_appTempl.h: 348

**Related method**

[FRAppGetDocOfPDDoc](#)

[FRAppSetActiveDocOfPDDoc](#)

**FRAppGetActiveTool****Syntax**

```
FR_Tool FRAppGetActiveTool (void );
```

**Description**

Gets the active tool from the application.

**Return**

The active tool.

**Head file reference**

fr\_appTempl.h: 242

**Related method**[FRAppSetActiveTool](#)[FRAppRegisterTool](#)**FRAppGetAdvWnd****Syntax**

```
HWND FRAppGetAdvWnd (void );
```

**Description**

Gets the window handle to show the advertisement.

**Return**

The window handle to show the advertisement.

**Head file reference**

fr\_appTempl.h: 1495

**Related method****Since**[SDK LATEEST VERSION > 2.1.0.4](#)**FRAppGetAppDataPath****Syntax**

```
FS_BOOL FRAppGetAppDataPath (  
    FS_WideString* outAppPath  
) ;
```

**Description**

Gets the execution file path of current running reader.

**Parameter**

---

outAppPath	[Out] The string buffer used to receive the path which Foxit Reader used to store some application's data. It will be filled automatically by reader.
------------	---

---

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

fr\_appTempl.h: 117

## FRAppGetAppTitle

### Syntax

```
FS_BOOL FRAppGetAppTitle (
    FS_WideString* outTitle
);
```

### Description

Gets the [FS\\_WideString](#) corresponding to the application's title.

### Parameter

---

outTitle	[Out] The string buffer used to receive the title of application. It will be filled by reader.
----------	--

---

### Return

[TRUE](#) for success, otherwise not.

### Head file reference

fr\_appTempl.h: 105

### Related method

[FRAppGetName](#)

**Note:** The user might have changed this, so do not use it to determine what the application is. Use [FRAppGetName](#) instead.

## FRAppGetCurMeasurementUnits

### Syntax

```
FS_INT32 FRAppGetCurMeasurementUnits (void );
```

### Description

Gets the current measurement unites.

### Return

The current measurement unites.

### Head file reference

fr\_appTempl.h: 1572

### Since

[SDK LATEEST VERSION > 3.0](#)

## FRAppGetCurrentChildFrm

### Syntax

---

```
HWND FRAppGetCurrentChildFrm (void );
```

**Description**

Gets the current child frame handle.

**Return**

The current child frame handle.

**Head file reference**

fr\_appTempl.h: 1856

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRAppGetCustomRecentFileListItemCount****Syntax**

```
FS_INT32 FRAppGetCustomRecentFileListItemCount (
    FS_LPCSTR lpsRecentFileName
);
```

**Description**

Gets the item count of the custom recent file list.

**Parameter**


---

lpsRecentFileName	[In] The input custom recent file list name.
-------------------	--

---

**Return**

The item count of the custom recent file list.

**Head file reference**

fr\_appTempl.h: 1432

**Related method**

[FRAppCreateCustomRecentFileList](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)

**FRAppGetDocOfPDDoc****Syntax**

```
FR_Document FRAppGetDocOfPDDoc (
    FS_INT32 index
```

);

**Description**

Gets the document view by specified index.

**Parameter**


---

index	[In] The index of specified document view with a <a href="#">FPD Document</a> object. The range is 0 to (CountDocsOfPDDoc()-1);
-------	---

---

**Return**

A document window.

**Head file reference**

fr\_appTempl.h: 335

**Related method**

[FRAppCountDocsOfPDDoc](#)

[FRAppGetActiveDocOfPDDoc](#)

**Note:** This method just return the document window which for displaying PDF file type, ignore other format file window.

**FRAppGetECMFileDialog****Syntax**

```
void* FRAppGetECMFileDialog (
    FS LPCSTR name,
    FR\_DRMSecurityCallbacks callbacks
);
```

**Description**

Registers a security handler. Invoked this interface to process the PDF documents that are encrypted by customer security handler. Since: [SDK LATEST VERSION](#) > 9.5

**Parameter**


---

name	[In] The name of the security handler.
------	--

---

callbacks	[In] The structure containing the security handler callback functions.
-----------	--

---

**Return**

void

**Head file reference**

---

fr\_appTempl.h: 2218

**Related method**[FRAppUnRegisterSecurityHandler](#)**FRAppGetEditionType****Syntax**

```
void FRAppGetEditionType (
    FS\_ByteString* outType
);
```

**Description**

Gets the edition type. There are five types, Business, Standard, Express, Enterprise and Free.

**Parameter**


---

outType	[Out] It receives the type value.
---------	-----------------------------------

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 1185

**Since**[SDK LATEEST VERSION > 1.0](#)**FRAppGetGEModule****Syntax**

```
void* FRAppGetGEModule (void );
```

**Description**

Gets the app GE module.

**Return**

The app GE module.

**Head file reference**

fr\_appTempl.h: 706

**FRAppGetKeyfileStartAndExpireDate****Syntax**

```
FS_BOOL FRAppGetKeyfileStartAndExpireDate (
    FS_WideString* outStartDate,
    FS_WideString* outExpireDate,
    FS_BOOL* outExpire
);
```

**Description**

Gets the start date and the expire date of the key file.

**Parameter**

---

outStartDate	[Out] It receives the start date.
--------------	-----------------------------------

---

outExpireDate	[Out] It receives the expire date.
---------------	------------------------------------

---

outExpire	[Out] Whether the key file expired or not.
-----------	--

---

**Return**

TRUE for success, otherwise the key file does not exist or is invalid.

**Head file reference**

fr\_appTempl.h: 1741

**Related method****Since**

[SDK LATEEST VERSION > 7.2.2](#)

**FRAppGetLocalFontName****Syntax**

```
void FRAppGetLocalFontName (
    FS_WideString* outFontName
);
```

**Description**

Gets the local font name.

**Parameter**

---

outFontName	[Out] It receives the local font name.
-------------	--

---

**Return**

The local font name.

**Head file reference**

fr\_appTempl.h: 1539

#### Related method

##### Since

[SDK LATEEST VERSION > 3.0](#)

## FRAppGetMainframeShow

#### Syntax

```
FS_BOOL FRAppGetMainframeShow (void );
```

#### Description

Checks whether the main frame will be shown or not when Foxit Reader starts up.

#### Return

Whether the main frame will be shown or not when Foxit Reader starts up.

#### Head file reference

fr\_appTempl.h: 1915

#### Related method

##### Since

[SDK LATEEST VERSION > 7.3.1](#)

## FRAppGetMainFrameWnd

#### Syntax

```
HWND FRAppGetMainFrameWnd (void );
```

#### Description

Gets the main frame window of Foxit Reader.

#### Return

A *HWND* object for windows.

#### Head file reference

fr\_appTempl.h: 490

## FRAppGetMainFrameWndByIndex

#### Syntax

```
void* FRAppGetMainFrameWndByIndex (
    FS_INT32 nIndex
);
```

**Description****Parameter**

nIndex	[In]
--------	------

**Return****Head file reference**

fr\_appTempl.h: 2153

**Since**[SDK\\_LATEEST\\_VERSION > 9.1](#)**FRAppGetMainFrameWndCount****Syntax**

FS\_INT32 FRAppGetMainFrameWndCount (void );

**Description****Return****Head file reference**

fr\_appTempl.h: 2144

**Since**[SDK\\_LATEEST\\_VERSION > 9.1](#)**FRAppGetMenuBar****Syntax**

FR\_MenuBar FRAppGetMenuBar (void );

**Description**

Gets Foxit Reader's menu bar.

**Return**

The menu bar.

**Head file reference**

fr\_appTempl.h: 170

**FRAppGetModelessParentWnd**

**Syntax**

```
HWND FRAppGetModelessParentWnd (void );
```

**Description****Return****Head file reference**

fr\_appTempl.h: 2163

**Since**

[SDK\\_LATEEST\\_VERSION > 9.1](#)

**FRAppGetModuleFileName****Syntax**

```
void FRAppGetModuleFileName (
    FS_WideString* outModuleFileName
);
```

**Description**

Gets the module file name of application.

**Parameter**

---

outModuleFileName	[Out] The string buffer used to receive the module file name of application.
-------------------	--

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 1205

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRAppGetModuleMgr****Syntax**

```
void* FRAppGetModuleMgr (void );
```

**Description**

Gets the app module manager.

**Return**

The app module manager.

**Head file reference**

fr\_appTempl.h: 714

**FRAppGetMousePos****Syntax**

```
FS_DevicePoint FRAppGetMousePos (void );
```

**Description**

Gets the mouse position. The mouse position is specified in screen coordinates.

**Return**

The [FS\\_DevicePoint](#) specified in screen coordinates.

**Head file reference**

fr\_appTempl.h: 421

**FRAppGetName****Syntax**

```
void FRAppGetName (
    FS\_ByteString* outName
);
```

**Description**

Gets the [FS\\_ByteString](#) corresponding to the application's name, which is the name of the file containing the Foxit Reader application.

**Parameter**

---

outName	[Out] The string buffer used to receive the name of application. It will be filled by reader.
---------	--

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 82

**Related method**[FRAppGetTitle](#)**FRAppGetNavPanelCount****Syntax**

```
FS_INT32 FRAppGetNavPanelCount (void );
```

**Description**

Gets the count of the navigation panel.

**Return**

The count of the navigation panel.

**Head file reference**

fr\_appTempl.h: 1607

**Related method**

[FRAppRegisterNavPanelView](#)

[FRAppDisableAllPanel](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

**FRAppGetNavPanelNameByIndex****Syntax**

```
void FRAppGetNavPanelNameByIndex (
    FS_INT32 nIndex,
    FS_BytString* outName
);
```

**Description**

Gets the name of the navigation panel.

**Parameter**

---

nIndex	[In] The input index of the navigation panel.
--------	---

---

outName	[Out] It receives the name of the navigation panel.
---------	---

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1594

**Related method**

[FRAppRegisterNavPanelView](#)

[FRAppDisableAllPanel](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

**FRAppGetNavPanelViewByName****Syntax**

```
HWND FRAppGetNavPanelViewByName (
    HWND hChildfrm,
    FS_LPCSTR lpsName
);
```

**Description**

Gets the window handle of the navigation panel view.

**Parameter**


---

<code>hChildfrm</code>	[In] The child frame the navigation panel belongs to. You can invoke <a href="#">FRDocGetDocFrameHandler</a> to get the child frame.
<code>lpsName</code>	[In] The input name of the navigation panel.

---

**Return**

The window handle of the navigation panel view.

**Head file reference**

`fr_appTempl.h: 1581`

**Related method**

[FRAppRegisterNavPanelView](#)  
[FRAppDisableAllPanel](#)

**Since**

[SDK LATEEST VERSION > 7.1.0.0](#)

**FRAppGetOEMVersion****Syntax**

```
void FRAppGetOEMVersion (
    FS_ByteString* outOEMVersion
);
```

**Description**

Gets the OEM version name.

**Parameter**


---

<code>outOEMVersion</code>	[Out] It receives the OEM version name. See <a href="#">FROEMVersion</a> .
----------------------------	--

---

**Return**

`void`

**Head file reference**

fr\_appTempl.h: 1452

**Since**[SDK LATEEST VERSION > 2.1.0.4](#)**FRAppGetPageContextMenuPos****Syntax**

```
FS_BOOL FRAppGetPageContextMenuPos (
    FS\_DevicePoint* outPoint
);
```

**Description**

Gets the position of context menu on the page view.

**Parameter**

---

outPoint	[Out] It receives the position.
----------	---------------------------------

---

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

fr\_appTempl.h: 1293

**Related method**[FRAppRegisterForContextMenuAddition](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FRAppGetPassphrase****Syntax**

```
void FRAppGetPassphrase (
    FS\_LPCSTR pstrEncryptedText,
    FS\_LPSTR* ppstrPlainText
);
```

**Description**

This interface is invoked by the plug-in to confirm that the host environment is Foxit Reader and is legal. The following steps show the procedure:

- 1. Plug-in generates a random string of 128 byte length and encrypts it using the public key.

- 2. Plug-in allocates one buffer of 128 bytes and calls [FRAppGetPassPhrase](#) () and passes the encrypted string.
- 3. Foxit reader will decrypt the encrypted text using the private key and copy it to the buffer *pstrPlainText* .
- 4. Plug-in matches the plain text against what it had generated.

**Parameter**


---

pstrEncryptedText	[In] An encrypted random string of 128 byte.
-------------------	--

---

ppstrPlainText	[Out] It receives the plain text from Foxit Reader.
----------------	---

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 477

**FRAppGetRecentFileIndex****Syntax**

```
FS_INT32 FRAppGetRecentFileIndex (
    FS_LPCWSTR lpszPathName,
    FS_BOOL bFolder
);
```

**Description**

Gets the index of the specified recent file.

**Parameter**


---

lpszPathName	[In] The recent opened file path.
--------------	-----------------------------------

---

bFolder	[In] Whether the path is a file or a folder.
---------	--

---

**Return**

The index of the specified recent file.

**Head file reference**

fr\_appTempl.h: 1934

**Related method**[FRAppGetRecentFileList](#)[FRAppAddFileToRecentFileList](#)[FRAppClearRecentFileList](#)[FRAppRemoveFileFromRecentFileList](#)

**Since**  
[SDK\\_LATEEST\\_VERSION > 8.1](#)

### **FRAppGetRecentFileList**

**Syntax**

```
void FRAppGetRecentFileList (
    FS\_WideStringArray* pArrFileList
);
```

**Description**

Gets the recent opened file list.

**Parameter**

---

pArrFileList	[Out] It receives the recent opened file list.
--------------	--

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 860

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

### **FRAppGetRecentFileSize**

**Syntax**

```
FS\_INT32 FRAppGetRecentFileSize (void );
```

**Description**

Gets the recent file list size.

**Return**

The recent file list size.

**Head file reference**

fr\_appTempl.h: 1195

**Related method**

[FRAppGetRecentFileList](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRAppGetRecentFolderListSize

### Syntax

```
FS_INT32 FRAppGetRecentFolderListSize (void );
```

### Description

Gets the recent folder list size.

### Return

The recent folder list size.

### Head file reference

fr\_appTempl.h: 1785

### Related method

[FRAppGetRecentFileList](#)

### Since

[SDK LATEEST VERSION > 7.3](#)

## FRAppGetRibbonBar

### Syntax

```
FR_RibbonBar FRAppGetRibbonBar (
    void* pParentWnd
);
```

### Description

Gets Foxit Reader's ribbon bar.

### Parameter

---

pParentWnd	[In] A pointer to the parent window. It represents the <i>MFC CWnd*</i> . Sets it to NULL as default. You can get the parent window through <a href="#">FRDocGetUIParentWnd</a> .
------------	---

---

### Return

The ribbon bar.

### Head file reference

fr\_appTempl.h: 966

### Since

[SDK LATEEST VERSION > 1.0](#)

## FRAppGetRibbonBar2

**Syntax**

```
FR_RibbonBar FRAppGetRibbonBar2 (
    HWND hWnd
);
```

**Description**

Gets Foxit Reader's ribbon bar.

**Parameter**

---

hWnd	[In] A handle to the parent window. It represents the <i>MFC HWND</i> . Sets it to NULL as default. You can get the parent window through <a href="#">FRDocGetUIParentWnd</a> .
------	---

---

**Return**

The ribbon bar.

**Head file reference**

fr\_appTempl.h: 2182

**Since**

[SDK LATEEST VERSION > 9.1](#)

**FRAppGetRibbonBarBackGroundColor****Syntax**

```
FS_COLORREF FRAppGetRibbonBarBackGroundColor (void );
```

**Description**

Gets the back ground color of ribbon bar.

**Return**

The back ground color of ribbon bar.

**Head file reference**

fr\_appTempl.h: 977

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRAppGetRibbonBarBtnBackGroundColor****Syntax**

```
FS_COLORREF FRAppGetRibbonBarBtnBackGroundColor (
    FS_BOOL bIsHighlighted,
    FS_BOOL bIsPressed
);
```

**Description**

Gets the back ground color of ribbon bar button.

**Parameter**

---

bIsHighlighted	[In] Whether the button is highlighted.
----------------	---

---

bIsPressed	[In] Whether the button is pressed.
------------	-------------------------------------

---

**Return**

The back ground color of ribbon bar button.

**Head file reference**

fr\_appTempl.h: 986

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRAppGetRibbonBaseBorderColor****Syntax**

```
FS_COLORREF FRAppGetRibbonBaseBorderColor (void );
```

**Description**

Gets the back ground color of ribbon base border.

**Return**

The back ground color of ribbon base border.

**Head file reference**

fr\_appTempl.h: 1776

**Since**

[SDK LATEEST VERSION > 7.2.2](#)

**FRAppGetRibbonCategoryColor****Syntax**

```
FS_COLORREF FRAppGetRibbonCategoryColor (void );
```

**Description**

Gets the back ground color of ribbon category.

**Return**

The back ground color of ribbon category.

**Head file reference**

fr\_appTempl.h: 1754

**Since**[SDK LATEEST VERSION > 7.2.2](#)**FRApGetRibbonElementColor****Syntax**

```
FS_BOOL FRApGetRibbonElementColor (
    FS_COLORREF* clrCheckButton,
    FS_COLORREF* clrCheckHighlight,
    FS_COLORREF* clrHightButton,
    FS_COLORREF* clrPressButton
);
```

**Description**

Gets the back ground color of ribbon element.

**Parameter**

---

clrCheckButton	[Out] The color when the button is selected.
----------------	--

---

clrCheckHighlight	[Out] The color when the mouse is over the selected button.
-------------------	---

---

clrHightButton	[Out] The color when the mouse is over the none-selected button.
----------------	--

---

clrPressButton	[Out] The color when the button is being clicked.
----------------	---

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1763

**Since**[SDK LATEEST VERSION > 7.2.2](#)**FRApGetRibbonTooltipBorderColor****Syntax**

```
FS_COLORREF FRApGetRibbonTooltipBorderColor (void );
```

**Description**

Gets the border color of the ribbon tooltip.

**Return**

The border color of the ribbon tooltip.

**Head file reference**

fr\_appTempl.h: 1925

**Since**

[SDK\\_LATEEST\\_VERSION > 8.0.2](#)

**FRApGetStartAppMode****Syntax**

```
FS_INT32 FRApGetStartAppMode (void );
```

**Description**

Gets the app starting mode, 0 for not open any documents, and 1 for opening some documents at the same time.

**Return**

The app starting mode, 0 for not open any documents, and 1 for opening some documents at the same time.

**Head file reference**

fr\_appTempl.h: 1618

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

**FRApGetStartMenuOfTabbedToobarMode****Syntax**

```
FR_Menu FRApGetStartMenuOfTabbedToobarMode (void );
```

**Description**

Gets the start menu which is in tabbed toolbar mode.

**Return**

A [FR\\_Menu](#) object indicates the start menu of tabbed toolbar mode. [NULL](#) if current toolbar mode is not tabbed mode.

**Head file reference**

fr\_appTempl.h: 758

**Note:** If a FR\_Menu is added into a exist menu in commonent toolbar mode, and need add into the start menu, call FRApIsRibbonMode() first, then call

FRAppGetStartMenuOfTabbedToobarMode() to get the start menu and add the menu item to start menu again.

## FRAppGetStatusBarBkGroundColor

### Syntax

```
FS_COLORREF FRAppGetStatusBarBkGroundColor (void );
```

### Description

Gets the back ground color of status bar in ribbon mode.

### Return

The back ground color of status bar.

### Head file reference

fr\_appTempl.h: 1638

### Since

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

## FRAppGetStatusBarBkGroundPath

### Syntax

```
void FRAppGetStatusBarBkGroundPath (
    FS_WideString* outPath
);
```

### Description

Gets the path of the back ground color picture of the status bar in classic mode.

### Parameter

---

outPath	[Out] It receives the path of the back ground color picture of the status bar in classic mode.
---------	--

---

### Return

void.

### Head file reference

fr\_appTempl.h: 1647

### Since

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

## FRAppGetTaskPaneWndHandleByDoc

**Syntax**

```
HWND FRAppGetTaskPaneWndHandleByDoc (
    FR\_Document frDoc
);
```

**Description**

Gets the window handle of the task pane.

**Parameter**


---

frDoc	[In] The document associated with the task pane.
-------	--

---

**Return**

The window handle of the task pane.

**Head file reference**

fr\_appTempl.h: 793

**Related method**

[FRAppRegisterTaskPaneView](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRAppGetToolBar****Syntax**

```
FRToolBar FRAppGetToolBar (
    FS\_INT32 index,
    void* pParentWnd
);
```

**Description**

Gets the specified toolbar.

**Parameter**


---

index	[In] The index of toolbar to obtain. The index range is 0 to ( <a href="#">FRAppCountToolbars</a> ()-1).
-------	--

---

pParentWnd	[In] A pointer to the parent window. It represents the <i>MFC CWnd*</i> . Sets it to NULL as default. You can get the parent window through <a href="#">FRDocGetUIParentWnd</a> .
------------	---

---

**Return**

The [FRToolBar](#) object.

**Head file reference**

fr\_appTempl.h: 140

**Related method**[FRAppCountToolbars](#)[FRAppGetToolBarByName](#)**FRAppGetToolBarByName****Syntax**

```
FRToolBar FRAppGetToolBarByName (
    FS_LPCSTR csName,
    void* pParentWnd
);
```

**Description**

Gets toolbar created with the specified name, All toolbars(including flyout toolbar) can be obtained by the method.

**Parameter**

---

csName	[In] The name of the toolbar.
pParentWnd	[In] A pointer to the parent window. It represents the <i>MFC CWnd*</i> . Sets it to NULL as default. You can get the parent window through <a href="#">FRDocGetUIParentWnd</a> .

---

**Return**

The [FRToolBar](#) , or [NULL](#) if no [FRToolBar](#) was found with the specified name.

**Head file reference**

fr\_appTempl.h: 153

**Related method**[FRToolBarSetName](#)**FRAppGetToolbarLocked****Syntax**

```
FS_BOOL FRAppGetToolbarLocked (void );
```

**Description**

Checks whether the toolbar is locked.

**Return**

[TRUE](#) means the toolbar is locked.

**Head file reference**

fr\_appTempl.h: 1215

**Related method**[FRAppSetToolbarLocked](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRAppGetToolByIndex****Syntax**

```
FR_Tool FRAppGetToolByIndex (
    FS_INT32 index
);
```

**Description**

Gets the specified tool.

**Parameter**

---

index	[In] The index of tool to obtain. The index range is 0 to ( <a href="#">FRAppCountTools</a> )-1).
-------	---

---

**Return**

The specified tool.

**Head file reference**

fr\_appTempl.h: 204

**Related method**[FRAppGetToolByName](#)[FRAppCountTools](#)**FRAppGetToolByName****Syntax**

```
FR_Tool FRAppGetToolByName (
    FS_LPCSTR csName
);
```

**Description**Gets the [FR\\_Tool](#) object that was registered under specified name.**Parameter**

---

csName	[In] The tool name.
--------	---------------------

---

**Return**

The tool that was registered under name, or [NULL](#) if no match was found.

**Head file reference**

fr\_appTempl.h: 199

**Related method**

[FRAppGetToolByIndex](#)

**FRAppGetUIParentWndByTaskPane****Syntax**

```
HWND FRAppGetUIParentWndByTaskPane (
    HWND taskPaneHwnd
);
```

**Description**

Gets the UI parent window handle.

**Parameter**

---

taskPaneHwnd	[In] The window handle of the task pane.
--------------	--

---

**Return**

The UI parent window handle.

**Head file reference**

fr\_appTempl.h: 1473

**Related method**

[FRAppRegisterTaskPaneView](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FRAppGetVersion****Syntax**

```
void FRAppGetVersion (
    FS\_WideString* outVersion
);
```

**Description**

Gets the [FS\\_WideString](#) corresponding to the application's version. The Format of version information is XX.XX.XX.XX(major number.minor number.maintainence number.build number).

**Parameter**

---

outVersion	[Out] The string buffer used to receive the version information of application. It will be filled by reader.
------------	--

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 94

**FRAppIsCurDocShowInBrowser****Syntax**

FS\_BOOL FRAppIsCurDocShowInBrowser (void );

**Description**

Tests whether the current document is opened in browser or not.

**Return**

[TRUE](#) for running in browser, otherwise [FALSE](#) .

**Head file reference**

fr\_appTempl.h: 1008

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRAppIsDisableAllPanel****Syntax**

FS\_BOOL FRAppIsDisableAllPanel (  
    [HWND](#) hFrameWnd  
);

**Description**

Checks whether all the panels are disabled.

**Parameter**

---

hFrameWnd	[In] The child frame window. Set it as NULL to use the current child frame window.
-----------	--

---

**Return**

[TRUE](#) means all the panels are disabled, otherwise not.

**Head file reference**

fr\_appTempl.h: 1282

**Related method**[FRAppDisableAllPanel](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRAppIsFipsMode****Syntax**

FS\_BOOL FRAppIsFipsMode (void );

**Description**

Whether the system is in Federal Information Processing Standard environment or not.

**Return**

TRUE if the system is in Federal Information Processing Standard environment.

**Head file reference**

fr\_appTempl.h: 2017

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 8.3.2](#)**FRAppIsFullScreen****Syntax**

FS\_BOOL FRAppIsFullScreen (void );

**Description**

Tests whether the application is running in full-screen mode.

**Return**[TRUE](#) if the application is currently in full-screen mode, otherwise not.**Head file reference**

fr\_appTempl.h: 387

**Related method**[FRAppShowFullScreen](#)[FRAppEndFullScreen](#)

**FRAppIsJSEnabled****Syntax**

```
FS_BOOL FRAppIsJSEnabled (void );
```

**Description**

Checks whether the javascript is enabled or not.

**Return**

TRUE if the javascript is enabled, otherwise FALSE.

**Head file reference**

fr\_appTempl.h: 1960

**Since**

[SDK LATEST VERSION > 8.1](#)

**FRAppIsLicenseValidOrInTrial****Syntax**

```
FS_BOOL FRAppIsLicenseValidOrInTrial (void );
```

**Description**

Checks whether the license is valid or is in trial.

**Return**

Checks whether the license is valid or is in trial.

**Head file reference**

fr\_appTempl.h: 1102

**Since**

[SDK LATEST VERSION > 1.0](#)

**FRAppIsMainfrmActivating****Syntax**

```
BOOL FRAppIsMainfrmActivating (  
    HWND hWnd  
);
```

**Description****Parameter**

---

hWnd	[In]
------	------

---

**Return****Head file reference**

fr\_appTempl.h: 2172

**Since**[SDK\\_LATEEST\\_VERSION > 9.1](#)**FRAppIsNeedCollectData****Syntax**

```
FS_BOOL FRAppIsNeedCollectData (void );
```

**Description**

Whether the cPDF data is need collected or not.

**Return**

TRUE if the cPDF data is need collected.

**Head file reference**

fr\_appTempl.h: 2007

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 8.2.1](#)**FRAppIsPDF2Doc****Syntax**

```
FS_BOOL FRAppIsPDF2Doc (  
    FPD_Document pSourDoc  
) ;
```

**Description**

whether it is 2.0 pdf document

**Parameter**

---

pSourDoc	[In] The specified document.
----------	------------------------------

---

**Return**

Non-zero means is 2.0 pdf document, otherwise means not.

**Head file reference**

fr\_appTempl.h: 2071

**Since**  
[SDK\\_LATEEST\\_VERSION > 9.1](#)

### FRAppIsReaderOnlyMode

#### Syntax

```
FS_BOOL FRAppIsReaderOnlyMode (  
    HWND hWnd  
);
```

#### Description

#### Parameter

---

hWnd	[In]
------	------

---

#### Return

#### Head file reference

fr\_appTempl.h: 2134

**Since**

[SDK\\_LATEEST\\_VERSION > 9.1](#)

### FRAppIsReadMode

#### Syntax

```
FS_BOOL FRAppIsReadMode (void );
```

#### Description

Checks whether Foxit Reader is in reading mode or not.

#### Return

[TRUE](#) if Foxit Reader is in reading mode, otherwise not.

#### Head file reference

fr\_appTempl.h: 1836

#### Related method

[FRAppExitReadMode](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

### FRAppIsRibbonMode

**Syntax**

```
FS_BOOL FRAppIsRibbonMode (void );
```

**Description**

Tests whether current toolbar mode is tabbed mode.

**Return**

[TRUE](#) if current toolbar mode is tabbed mode, otherwise [FALSE](#).

**Head file reference**

fr\_appTempl.h: 750

**FRAppIsRTLLanguage****Syntax**

```
FS_BOOL FRAppIsRTLLanguage (void );
```

**Description**

Checks whether the current language of app is right-to-left or not.

**Return**

TRUE if the current language of app is right-to-left, otherwise FALSE.

**Head file reference**

fr\_appTempl.h: 1443

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRAppIsRunEmbedded****Syntax**

```
FS_BOOL FRAppIsRunEmbedded (void );
```

**Description**

Tests whether the Foxit Reader is running embedded or not.

**Return**

[TRUE](#) for running embedded, otherwise [FALSE](#).

**Head file reference**

fr\_appTempl.h: 733

**FRAppLoadLibrary****Syntax**

---

```
void* FRAppLoadLibrary (
    FS\_LPCWSTR lpwsFileName,
    FRDirectoryType nDirectoryType
);
```

**Description**

Load the library.

**Parameter**


---

lpwsFileName	[In] Specifies the file name of library.
nDirectoryType	[In] The directory type.

---

**Return**

The library handle.

**Head file reference**

fr\_appTempl.h: 997

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRAppModalWindowIsOpen****Syntax**

```
FS_BOOL FRAppModalWindowIsOpen (void );
```

**Description**

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

**Return**

[TRUE](#) if a modal window is open, otherwise not.

**Head file reference**

fr\_appTempl.h: 429

**FRAppOnCmdMsgByName****Syntax**

```
FS_BOOL FRAppOnCmdMsgByName (
    FS\_LPCSTR lpsName,
```



---

```
FS_INT32 nCode,
void* pExtra,
void* pHandlerInfo
);
```

**Description**

Routes and dispatches command messages from plug-ins to plug-ins

**Parameter**


---

lpsName	[In] The name of menu or toolbar whose command messages need to be routed and dispatched.
nCode	[In] References to <i>MFC CCmdTarget::OnCmdMsg</i> .
pExtra	[In] References to <i>MFC CCmdTarget::OnCmdMsg</i> .
pHandlerInfo	[In] It represents the <i>MFC</i> struct <i>AFX_CMDHANDLERINFO</i> .

---

**Return**

Nonzero if the message is handled; otherwise 0.

**Head file reference**

fr\_appTempl.h: 1304

**Related method**

[FRAppRegisterCmdMsgEventHandler](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRAppOpenFileAttachment****Syntax**

```
FS_BOOL FRAppOpenFileAttachment (
    FPD\_Object pDict
);
```

**Description**

open a embed file.

**Parameter**


---

pDict	[In] The FileAttachment dictionary. param[in]: nPage The Page index. param[in]: bAnnot Is FileAttachment Annot.
-------	---

---

**Return**

Non-zero means succeed, otherwise failure.

**Head file reference**

fr\_appTempl.h: 2059

**Since**

[SDK\\_LATEEST\\_VERSION > 9.5](#)

**FRAppPopDocPropertyPage****Syntax**

```
void FRAppPopDocPropertyPage (
    FS\_LPCWSTR PageShowName
);
```

**Description**

Pops up the doc property pages.

**Parameter**

---

PageShowName	[In] The property page you want to display.
--------------	---

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 899

**Related method**

[FRAppRegisterDocPropertyPageHandler](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRAppRedrawRecentFileList****Syntax**

```
FS_BOOL FRAppRedrawRecentFileList (void );
```

**Description**

Redraws the recent file list in Ribbon mode.

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1358

**Related method**[FRAppGetRecentFileList](#)  
[FRAppAddFileToRecentFileList](#)  
[FRAppClearRecentFileList](#)**Since**[SDK LATEEST VERSION > 2.1.0.3](#)**FRAppRefreshLayerPanelView****Syntax**

```
void FRAppRefreshLayerPanelView (
    FPD Document pCpdfDoc
);
```

**Description****Parameter**

---

pCpdfDoc	[In]
----------	------

---

**Return****Head file reference**

fr\_appTempl.h: 2193

**Since**[SDK LATEEST VERSION > 9.1](#)**FRAppRegisterActionHandler****Syntax**

```
FS_BOOL FRAppRegisterActionHandler (
    FR ActionHandlerCallbacks callbacks
);
```

**Description**

Registers an action handler. The callbacks are called by Foxit Reader when the Foxit Reader will do the actions. You can implement your own process to customize the action process.

**Parameter**

---

callbacks	[In] The callbacks for action handler.
-----------	--

---

**Return**

TRUE means successful, otherwise not.

**Head file reference**

fr\_appTempl.h: 1347

**Since**

SDK\_LATEEST\_VERSION > 2.1

**FRApplRegisterAppEventHandler****Syntax**

```
FS_BOOL FRApplRegisterAppEventHandler (
    FR_AppEventCallbacks appEventCallbacks
);
```

**Description**

Registers a user-supplied procedure set to call when some application level event occurs.

**Parameter**

---

appEventCallbacks	[In] The callback set. Reader will call a corresponding callback when the app event occurs.
-------------------	---

---

**Return**

TRUE for success, otherwise not.

**Head file reference**

fr\_appTempl.h: 321

**FRApplRegisterCaptureHandler****Syntax**

```
FS_BOOL FRApplRegisterCaptureHandler (
    FR_CaptureCallbacks callbacks
);
```

**Description**

Registers the capture handler.

**Parameter**

---

callbacks	[In] A structure containing the capture handler's callback functions.
-----------	---

---

**Return**

[TRUE](#) means successful, otherwise not.

#### **Head file reference**

fr\_appTempl.h: 561

### **FRAppRegisterCmdMsgEventHandler**

#### **Syntax**

```
void* FRAppRegisterCmdMsgEventHandler (
    FR\_CmdMsgEventCallbacks callbacks
);
```

#### **Description**

Registers a callbacks called by the Foxit Reader to route and dispatch command messages and to handle the update of command user-interface objects, such as menu, toolbar.

#### **Parameter**

---

callbacks	[In] The input callbacks.
-----------	---------------------------

---

#### **Return**

The cmd msg event handler can be used to unregister by [FRAppUnRegisterCmdMsgEventHandler](#).

#### **Head file reference**

fr\_appTempl.h: 1237

#### **Related method**

[FRAppUnRegisterCmdMsgEventHandler](#)

#### **Since**

[SDK LATEEST VERSION > 2.0](#)

### **FRAppRegisterContentProvider**

#### **Syntax**

```
void FRAppRegisterContentProvider (
    FR\_ContentProviderCallbacks contentProviderCallbacks
);
```

#### **Description**

Registers a content provider so that the plug-in can process the protected document and provide decrypted document data.

#### **Parameter**

---

contentProviderCallbacks	[In] The content provider callbacks.
--------------------------	--------------------------------------

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 480

**FRAppRegisterDataCollectionHadler****Syntax**

```
void FRAppRegisterDataCollectionHadler (
    FR\_DataCollectionHandlerCallbacks callbacks
);
```

**Description**

Registers the data collection handler to collect the data as you need.

**Parameter**

---

callbacks	[In] The callbacks for data collection handler.
-----------	---

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 1462

**Related method**

[FRAppCollectNormalData](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FRAppRegisterDocHandlerOfPDDoc****Syntax**

```
FS_BOOL FRAppRegisterDocHandlerOfPDDoc (
    FR\_DocEventCallbacks docEventCallbacks
);
```

**Description**

Registers a user-supplied event handler to document window.

**Parameter**

---

docEventCallbacks	[In] The callback set. Reader will call a corresponding callback when the doc event occurs.
-------------------	---

---

**Return**

TRUE means successful, otherwise not.

**Head file reference**

fr\_appTempl.h: 377

**Note:** The callbacks of FR Doc Event Handler just be called while the document window is displaying PDF file type.

**FRAppRegisterDocPropertyPageHandler****Syntax**

```
FS_BOOL FRAppRegisterDocPropertyPageHandler (
    FR_DocPropertypageCallbacks docProperPageCallbacks
);
```

**Description**

Registers the event notification to document properties dialog.

**Parameter**

---

docProperPageCallbacks	[In] The callback set. Reader will call a corresponding callback when the document properties dialog is to be show or to be destroy.
------------------------	--

---

**Return**

TRUE for success, otherwise FALSE.

**Head file reference**

fr\_appTempl.h: 285

**Related method**

[FRAppAddDocPropertyPage](#)

**Note:** You must also call FRAppAddDocPropertyPage() in the FRDocPropertyPageOnCreate() callback to add a property page to Reader document properties dialog.

**FRAppRegisterExtraPrintInfoProvider****Syntax**

```
void FRAppRegisterExtraPrintInfoProvider (
    FR_ExtraPrintInfoProviderCallbacks callbacks
);
```

**Description**

Register the printing notifies.

**Parameter**

---

callbacks	[In] The callbacks which will be broadcast.
-----------	---

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 741

**FRAppRegisterForContextMenuAddition****Syntax**

```
FRAppRegisterForContextMenuAddition (
    menuName,
    proc,
    clientData
);
```

**Description**

Registers a user-supplied procedure to call after a context menu has been created but before it is shown to the user. The callback can add menu items to or remove menu items from the menu. \*\*\*\*\*

**Parameter**

---

[In] The name of the context menu to modify. Its names can be one of the following:

- menuName
- Name - Description
  - Page - The standard context menu for an FR\_PageView.
  - Select - The context menu for selected text.
  - Snapshot - The context menu for snapshot of FR\_PageView.
- 

---

proc	[In] The user-supplied procedure to call.
------	---

---

---

clientData	[In] A pointer to user-supplied data to pass to the procedure each time it is called.
------------	---

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_appTempl.h: 522

## FRAppRegisterMousePtHandler

### Syntax

```
FS_BOOL FRAppRegisterMousePtHandler (
    FR\_MousePtCallbacks callbacks
);
```

### Description

Registers the mouse point handler.

### Parameter

---

callbacks	[In] A structure containing the mouse point handler's callback functions.
-----------	---

---

### Return

[TRUE](#) means successful, otherwise not.

### Head file reference

fr\_appTempl.h: 59

### Related method

[FRToolSetAssociatedMousePtHandlerType](#)

## FRAppRegisterNavPanelView

### Syntax

```
FS_BOOL FRAppRegisterNavPanelView (
    FR\_PanelViewCallbacks panelViewCallbacks
);
```

### Description

Registers a navigation panel view into the Foxit Reader.

### Parameter

---

panelViewCallbacks	[In] A panel page callback set. When the navigation panel occurs a event, reader will call a corresponding callback to respond.
--------------------	---

---

### Return

[TRUE](#) for success, otherwise not.

### Head file reference

fr\_appTempl.h: 311

## FRAppRegisterOwnerFileType

### Syntax

```
FS_BOOL FRAppRegisterOwnerFileType (
    FR\_OwnerFileTypeHandlerCallbacks callbacks
);
```

### Description

Adds a file type to Foxit Reader. You can control the process of owner file type, such as opening, saving, sending email and so on.

### Parameter

---

callbacks	[In] The callback set for the owner file type handler.
-----------	--

---

### Return

[TRUE](#) for success, otherwise failure.

### Head file reference

fr\_appTempl.h: 931

### Since

[SDK LATEST VERSION > 1.0](#)

## FRAppRegisterPageHandlerOfPDDoc

### Syntax

```
void FRAppRegisterPageHandlerOfPDDoc (
    FR\_PageEventCallbacks callbacks
);
```

### Description

Registers a page-level event callback set.

### Parameter

---

callbacks	[In] The callback set. Reader will call a corresponding callback when the page event occurs.
-----------	--

---

### Return

void

### Head file reference

fr\_appTempl.h: 513

## FRAppRegisterPDFAPluginHandler

### Syntax

```
FS_BOOL FRAppRegisterPDFAPluginHandler (
    FR\_PDFAPPluginHandlerCallbacks pdfaPluginHandlerCallbacks
);
```

### Description

The callbacks of pdfa Handler just be called while the document was saved to pdfa.

### Parameter

---

pdfaPluginHandlerCallbacks	[In] The callback set. Reader will call a corresponding callback when save to pdfa occurs.
----------------------------	--

---

### Return

[TRUE](#) means successful, otherwise not.

### Head file reference

fr\_appTempl.h: 2097

### Since

[SDK LATEEST VERSION > 9.1](#)

## FRAppRegisterPOEventHandler

### Syntax

```
FS_BOOL FRAppRegisterPOEventHandler (
    FR\_POEventHandlerCallbacks callbacks
);
```

### Description

Registers a callbacks called by the Foxit Reader to do the page organizing, such as adding pages, deleting pages, replacing pages and so on.

### Parameter

---

callbacks	[In] The input callbacks.
-----------	---------------------------

---

### Return

TRUE for success, otherwise not.

### Head file reference

fr\_appTempl.h: 1516

### Related method

### Since

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)**FRAppRegisterPreferencePageHandler****Syntax**

```
FS_BOOL FRAppRegisterPreferencePageHandler (
    FR\_PreferPageCallbacks preferPageCallbacks
);
```

**Description**

Registers the event notification to preference dialog.

**Parameter**

---

preferPageCallbacks	[In] The callback set. Reader will call a corresponding callback when the preference dialog is to be show or to be destroy.
---------------------	---

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 257

**Related method**

[FRAppAddPreferencePage](#)  
[FRAppShowPreferenceDlg](#)

**Note:** You must also call FRAppAddPreferencePage() in the FRPrefPageOnCreate() callback to add a preference page to Reader preference dialog.

**FRAppRegisterRibbonRecentFileEventHandler****Syntax**

```
void* FRAppRegisterRibbonRecentFileEventHandler (
    FR\_RibbonRecentFileEventCallbacks callbacks
);
```

**Description**

Registers a callbacks set for Ribbon recent file list event handler.

**Parameter**

---

callbacks	[In] The input callbacks for Ribbon recent file list event handler.
-----------	---

---

**Return**

The pointer to Ribbon recent file list event handler can be destroyed by [FRAppDestroyRibbonRecentFileEventHandler](#) .

#### **Head file reference**

fr\_appTempl.h: 1681

#### **Related method**

[FRAppDestroyRibbonRecentFileEventHandler](#)

#### **Since**

[SDK LATEEST VERSION > 7.2.2](#)

## FRAppRegisterSecurityHandler

#### **Syntax**

```
void FRAppRegisterSecurityHandler (
    FS_LPCSTR name,
    FR_SecurityCallbacks callbacks
);
```

#### **Description**

Registers a security handler. Invoked this interface to process the PDF documents that are encrypted by customer security handler.

#### **Parameter**

name	[In] The name of the security handler.
callbacks	[In] The structure containing the security handler callback functions.

#### **Return**

void

#### **Head file reference**

fr\_appTempl.h: 441

#### **Related method**

[FRAppUnRegisterSecurityHandler](#)

## FRAppRegisterSecurityMethod

#### **Syntax**

```
FS_BOOL FRAppRegisterSecurityMethod (
    FR_SecurityMethodCallbacks cllbcks
);
```

**Description**

Registers the security method that you can manage your security method.

**Parameter**


---

cllbcks	[In] The callback set for security method.
---------	--

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 889

**Since**

[SDK LATEST VERSION > 1.0](#)

**FRApplRegisterSelectionHandler****Syntax**

```
FS_BOOL FRApplRegisterSelectionHandler (
    FR\_SelectionCallbacks callbacks
);
```

**Description**

Registers a new selection handler. Selection handlers allow the selection of items other than those that can be selected in the as-shipped Foxit viewer. For example, a selection handler could allow a user to select a sampled image. This method can be used to replace an existing selection handler that handles the same selection type.

**Parameter**


---

callbacks	[In] A structure containing the selection handler's callback functions. This structure must not be freed after calling <a href="#">FRApplRegisterSelectionHandler</a> ().
-----------	---

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_appTempl.h: 539

**Note:** Call [FRApplUnRegisterSelectionHandler](#) to unregister the selection handler and release the memory.

**FRApplRegisterTaskPaneView****Syntax**

```
FS_BOOL FRApplRegisterTaskPaneView (
```

```
    callbacks  
);
```

**Description**

Registers callback set for a task pane view.

**Parameter**

---

callbacks	[In] The callback set for a task pane view.
-----------	---

---

**Return**

[TRUE](#) for success, otherwise [FALSE](#).

**Head file reference**

fr\_appTempl.h: 769

**Related method**

[FRAppShowTaskPane](#)

**Since**

[SDK LATEEST VERSION > 1.0](#)

## FRAppRegisterTool

**Syntax**

```
void FRAppRegisterTool (  
    FR\_Tool tool  
);
```

**Description**

Registers a tool into the Foxit Reader.

**Parameter**

---

tool	[In] The <a href="#">FR_Tool</a> object containing the tool's callbacks. This object must not be freed after calling <a href="#">FRAppRegisterTool</a> (), but must be remained.
------	--

---

**Return****Head file reference**

fr\_appTempl.h: 29

**Related method**

[FRToolNew](#)

[FRAppSetActiveTool](#)

## FRAppRegisterWinMSGHandler

### Syntax

```
FS_BOOL FRAppRegisterWinMSGHandler (
    FR\_WinMSGCallbacks callbacks
);
```

### Description

Registers callback set for a windows MSG handler.

### Parameter

---

callbacks	[In] The callback set for a windows MSG handler.
-----------	--

---

### Return

[TRUE](#) for success, otherwise [FALSE](#).

### Head file reference

fr\_appTempl.h: 850

### Since

[SDK LATEEST VERSION > 1.0](#)

## FRAppRegisterWndProvider

### Syntax

```
void* FRAppRegisterWndProvider (
    FR\_WndProviderCallbacks callbacks
);
```

### Description

Registers a window provider to create a window above the document view. There are already some windows above the document view, such as PDF view, text view, ruler view.

### Parameter

---

callbacks	[In] The input callbacks used to control the window, it tells you when to create the window handle, when to show the window, when to scroll the window, and so on.
-----------	--

---

### Return

The returned value is reserved.

### Head file reference

fr\_appTempl.h: 1160

### Related method

[FRDocSetCurrentWndProvider](#)

[FRDocGetCurrentWndProvider](#)  
[FRAppUnRegisterWndProvider](#)

**Since**  
[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRApplRegistryGetBinary

### Syntax

```
FS_BOOL FRApplRegistryGetBinary (
    FS_LPCWSTR lpszSection,
    FS_LPCWSTR lpszEntry,
    unsigned char** ppData,
    unsigned int* pBytes
);
```

### Description

Gets the binary value in the specified section.

### Parameter

---

lpszSection	[In] The input registry section.
-------------	----------------------------------

---

lpszEntry	[In] The input registry entry.
-----------	--------------------------------

---

ppData	[Out] It receives the binary data.
--------	------------------------------------

---

pBytes	[Out] It receives the size of the binary data.
--------	--

---

### Return

[TRUE](#) means successful, otherwise not.

### Head file reference

fr\_appTempl.h: 644

## FRApplRegistryGetEntryCount

### Syntax

```
FS_INT32 FRApplRegistryGetEntryCount (
    FS_LPCWSTR lpszSection
);
```

### Description

Gets the entry count in the specified section.

**Parameter**


---

IpszSection	[In] The input registry section.
-------------	----------------------------------

---

**Return**

The entry count in the specified section.

**Head file reference**

fr\_appTempl.h: 667

**FRAppRegistryGetEntryName****Syntax**

```
FS_BOOL FRAppRegistryGetEntryName (
    FS_LPCWSTR IpszSection,
    FS_INT32 nIndex,
    FS_WideString* outName
);
```

**Description**

Gets the entry name in the specified section.

**Parameter**


---

IpszSection	[In] The input registry section.
-------------	----------------------------------

---



---

nIndex	[In] The specified index.
--------	---------------------------

---



---

outName	[Out] It receives the entry name.
---------	-----------------------------------

---

**Return**

TRUE means successful, otherwise not.

**Head file reference**

fr\_appTempl.h: 676

**FRAppRegistryGetInt****Syntax**

```
UINT FRAppRegistryGetInt (
    FS_LPCWSTR IpszSection,
    FS_LPCWSTR IpszEntry
);
```

**Description**

Gets the int value in the specified section.

**Parameter**

---

lpszSection	[In] The input registry section.
lpszEntry	[In] The input registry entry.

---

**Return**

The int value in the specified section.

**Head file reference**

fr\_appTempl.h: 634

**FRAppRegistryGetKeyCounts****Syntax**

```
FS_INT32 FRAppRegistryGetKeyCounts (
    FS\_LPCWSTR lpszSection
);
```

**Description**

Gets the counts of key for the specified section.

**Parameter**

---

lpszSection	[In] The input registry section.
-------------	----------------------------------

---

**Return**

The counts of key for the specified section.

**Head file reference**

fr\_appTempl.h: 942

**Related method**

[FRAppRegistryGetProfilePath](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRAppRegistryGetKeyName****Syntax**

```
FS_BOOL FRAppRegistryGetKeyName (
    FS\_LPCWSTR lpszSection,
    FS\_INT32 nIndex,
    FS\_WideString* outName
);
```

**Description**

Gets the key name in the specified section.

**Parameter**

lpszSection	[In] The input registry section.
nIndex	[In] The specified index.
outName	[Out] It receives the key name.

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_appTempl.h: 953

**Related method**

[FRAppRegistryGetProfilePath](#)

**Since**

[SDK LATEEST VERSION > 1.0](#)

[FRAppRegistryGetProfilePath](#)

**Syntax**

```
void FRAppRegistryGetProfilePath (
    FS\_WideString* outProfilePath
);
```

**Description**

Gets the registry's profile path of the Foxit Reader.

**Parameter**

outProfilePath	[Out] It receives the registry's profile path of the Foxit Reader.
----------------	--

**Return**

void

**Head file reference**

fr\_appTempl.h: 579

## FRAppRegistryGetString

### Syntax

```
FS_BOOL FRAppRegistryGetString (
    FS_LPCWSTR lpszSection,
    FS_LPCWSTR lpszEntry,
    FS_WideString* outString
);
```

### Description

Gets the string value in the specified section.

### Parameter

lpszSection	[In] The input registry section.
lpszEntry	[In] The input registry entry.
outString	[Out] It receives the string.

### Return

[TRUE](#) means successful, otherwise not.

### Head file reference

fr\_appTempl.h: 656

## FRAppRegistryRemoveEntry

### Syntax

```
void FRAppRegistryRemoveEntry (
    FS_LPCWSTR lpszSection,
    FS_LPCWSTR lpszEntry
);
```

### Description

Removes the specified entry.

### Parameter

lpszSection	[In] The input registry section.
lpszEntry	[In] The input registry entry.

### Return

void

**Head file reference**

fr\_appTempl.h: 696

**FRAppRegistryRemoveSection****Syntax**

```
void FRAppRegistryRemoveSection (
    FS_LPCWSTR lpszSection
);
```

**Description**

Removes the specified section.

**Parameter**

---

lpszSection	[In] The input registry section.
-------------	----------------------------------

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 687

**FRAppRegistryWriteBinary****Syntax**

```
FS_BOOL FRAppRegistryWriteBinary (
    FS_LPCWSTR lpszSection,
    FS_LPCWSTR lpszEntry,
    FS_LPBYTE pData,
    nValue
);
```

**Description**

Write binary value to the specified registry. If you want to set the value in registry like "*HKEY\_CURRENT\_USER\Software\Foxit Software\Foxit Reader\plugins\Create Link*" , *lpszSection* can be set as "*Create Link*" . If you want to set the value in registry like "*HKEY\_CURRENT\_USER\Software\Foxit Software\Foxit Reader\plugins\Create Link\Annot*" *lpszSection* can be set as "*Create Link\Annot*" .

**Parameter**

---

lpszSection	[In] The input registry section.
-------------	----------------------------------

---

lpszEntry	[In] The input registry entry.
-----------	--------------------------------

---

---

pData	[In] The input binary data.
-------	-----------------------------

---

nValue	[In] The size of the binary data.
--------	-----------------------------------

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_appTempl.h: 603

**FRAppRegistryWriteInt****Syntax**

```
FS_BOOL FRAppRegistryWriteInt (
    FS_LPCWSTR lpszSection,
    FS_LPCWSTR lpszEntry,
    FS_INT32 nValue
);
```

**Description**

Write int value to the specified registry. If you want to set the value in registry like "*HKEY\_CURRENT\_USER\Software\Foxit Software\Foxit Reader\plugins\Create Link*" , *lpszSection* can be set as "*Create Link*" . If you want to set the value in registry like "*HKEY\_CURRENT\_USER\Software\Foxit Software\Foxit Reader\plugins\Create Link\Annot\*" *lpszSection* can be set as "*Create Link\Annot*" .

**Parameter**


---

lpszSection	[In] The input registry section.
-------------	----------------------------------

---

lpszEntry	[In] The input registry entry.
-----------	--------------------------------

---

nValue	[In] The input int value.
--------	---------------------------

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_appTempl.h: 588

**FRAppRegistryWriteString****Syntax**

```
FS_BOOL FRAppRegistryWriteString (
    FS_LPCWSTR lpszSection,
    FS_LPCWSTR lpszEntry,
    FS_LPCWSTR lpszValue
```

---

);

**Description**

Write string to the specified registry. If you want to set the value in registry like "*HKEY\_CURRENT\_USER\Software\Foxit Software\Foxit Reader\plugins\Create Link*" , *lpszSection* can be set as "*Create Link*" . If you want to set the value in registry like "*HKEY\_CURRENT\_USER\Software\Foxit Software\Foxit Reader\plugins\Create Link\Annot*" *lpszSection* can be set as "*Create Link\Annot*" .

**Parameter**

<i>lpszSection</i>	[In] The input registry section.
<i>lpszEntry</i>	[In] The input registry entry.
<i>lpszValue</i>	[In] The input string.

**Return**

TRUE means successful, otherwise not.

**Head file reference**

fr\_appTempl.h: 619

**FRAppReleaseNavPanelView****Syntax**

```
void FRAppReleaseNavPanelView (
    void* pNavPanelView
);
```

**Description**

Releases the navigation panel view returned by [FRAppAddNavPanelView](#) .

**Parameter**

<i>pNavPanelView</i>	[In] Returned by <a href="#">FRAppAddNavPanelView</a> .
----------------------	---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 2031

**Since**

[SDK\\_LATEEST\\_VERSION > 8.3.2](#)

## FRAppReLoadStartPage

### Syntax

```
void FRAppReLoadStartPage (
    FS\_LPCWSTR lpwsPath
);
```

### Description

Reloads the start page.

### Parameter

---

lpwsPath	[In] The input path of the start page.
----------	--

---

### Return

void.

### Head file reference

fr\_appTempl.h: 1550

### Related method

### Since

[SDK\\_LATEEST\\_VERSION > 3.0](#)

## FRAppRemoveFileFromCustomRecentFileList

### Syntax

```
FS_BOOL FRAppRemoveFileFromCustomRecentFileList (
    FS\_LPCSTR lpsRecentFileName,
    FS\_LPCWSTR lpwsFilePath
);
```

### Description

Removes the recent file path from the specified recent file list.

### Parameter

---

lpsRecentFileName	[In] The name that specifies the custom recent file list from which the recent file path will be removed.
-------------------	---

---

---

lpwsFilePath	[In] The recent file path that will be removed.
--------------	---

---

### Return

[TRUE](#) for success, otherwise failure.

### Head file reference

fr\_appTempl.h: 1044

**Related method**

[FRAppCreateCustomRecentFileList](#)

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRAppRemoveFileFromRecentFileList**

**Syntax**

```
void FRAppRemoveFileFromRecentFileList (
    FS_INT32 nIndex
);
```

**Description**

Removes the recent opened file path from the list.

**Parameter**

---

nIndex	[In] The index of recent opened file path to be removed.
--------	--

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1017

**Related method**

[FRAppGetRecentFileList](#)  
[FRAppAddFileToRecentFileList](#)  
[FRAppClearRecentFileList](#)

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRAppRemoveTaskPanelByName**

**Syntax**

```
FS_BOOL FRAppRemoveTaskPanelByName (
    FS_LPSTR nameOfPaneView
);
```

**Description**

Removes the specified task panel.

**Parameter**

---

nameOfPaneView	[In] The input name of task panel to be removed.
----------------	--

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1259

**Related method**

[FRAppRegisterTaskPaneView](#)

[FRAppShowTaskPane](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRAppRestartProcess****Syntax**

```
void FRAppRestartProcess (
    FS\_BOOL bReOpenFile,
    FS\_BOOL bDelay
);
```

**Description**

Restarts the process of Foxit Reader or Foxit PhantomPDF.

**Parameter**


---

bReOpenFile	[In] Whether to reopen the files or not.
-------------	--

---

bDelay	[In] Whether to delay the request or not.
--------	---

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1627

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

**FRAppSetActiveDocOfPDDoc****Syntax**

```
FS\_BOOL FRAppSetActiveDocOfPDDoc (
    FR\_Document doc
);
```

**Description**

Sets a document view to top-most.

**Parameter**

---

doc	[In] The <a href="#">FR_Document</a> object.
-----	--

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_appTempl.h: 360

**Related method**

[FRAppGetActiveDocOfPDDoc](#)

[FRAppCountDocsOfPDDoc](#)

[FRAppGetDocOfPDDoc](#)

**FRAppSetActiveTool****Syntax**

```
FS_BOOL FRAppSetActiveTool (
    FR\_Tool tool,
    FS_BOOL persistent
);
```

**Description**

Sets the active tool. It does nothing if the specified tool is not currently enabled. The [FRToolIsEnabled\(\)](#) callback in [FR\\_ToolCallbacksRec](#) structure determines whether a tool is enabled. If this callback is [NULL](#), the tool is always enabled.

**Parameter**

---

tool	[In] The tool to set as the active tool.
------	--

---

persistent	[In] A flag that indicates a preference as to whether the tool stays active after it is used. <a href="#">TRUE</a> is a hint that the tool should, if possible, stay active for an arbitrary number of operations (whatever that happens to be) rather than doing a one shot operation and restoring the prior active tool.
------------	---

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_appTempl.h: 194

**Related method**[FRAppRegisterTool](#)[FRAppGetActiveTool](#)**FRAppSetCustomRecentFileListItemBitmap****Syntax**

```
FS_BOOL FRAppSetCustomRecentFileListItemBitmap (
    FS_LPCSTR lpsRecentFileName,
    FS_DIBitmap pItemBitmap
);
```

**Description**

Sets the icon that will be displayed in front of the recent file list label.

**Parameter**

---

lpsRecentFileName	[In] The input custom recent file list name.
-------------------	--

---

pItemBitmap	[In] The icon that will be displayed in front of the recent file list label.
-------------	--

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1382

**Related method**[FRAppCreateCustomRecentFileList](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)**FRAppSetCustomRecentFileListItemTitle****Syntax**

```
FS_BOOL FRAppSetCustomRecentFileListItemTitle (
    FS_LPCSTR lpsRecentFileName,
    FS_INT32 nIndex,
    FS_LPCWSTR lpwsFileTitle
);
```

**Description**

Sets the title of the recent file list item.

**Parameter**


---

lpsRecentFileName	[In] The input custom recent file list name.
-------------------	--

---

nIndex	[In] The index of the recent file list item.
--------	--

---

lpwsFileTitle	[In] The title of the recent file list item.
---------------	--

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1406

**Related method**

[FRAppCreateCustomRecentFileList](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.3](#)

**FRAppSetCustomRecentFileListItemTooltip****Syntax**

```
FS_BOOL FRAppSetCustomRecentFileListItemTooltip (
    FS_LPCSTR lpsRecentFileName,
    FS_INT32 nIndex,
    FS_LPCWSTR lpwsFileTooltip
);
```

**Description**

Sets the tooltip of the recent file.

**Parameter**


---

lpsRecentFileName	[In] The input custom recent file list name.
-------------------	--

---

nIndex	[In] The index of the recent file list item.
--------	--

---

lpwsFileTooltip	[In] The tooltip of the recent file.
-----------------	--------------------------------------

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1419

**Related method**[FRAppCreateCustomRecentFileList](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)**FRAppSetCustomRecentFileListLabel****Syntax**

```
FS_BOOL FRAppSetCustomRecentFileListLabel (
    FS_LPCSTR lpsRecentFileName,
    FS_LPCWSTR lpwsLabel
);
```

**Description**

Sets the label of the custom recent file list.

**Parameter**

---

lpsRecentFileName	[In] The input custom recent file list name.
-------------------	--

---

lpwsLabel	[In] The input label of recent file list.
-----------	---

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1370

**Related method**[FRAppCreateCustomRecentFileList](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)**FRAppSetCustomRecentFileListMaxSize****Syntax**

```
FS_BOOL FRAppSetCustomRecentFileListMaxSize (
    FS_LPCSTR lpsRecentFileName,
    FS_INT32 nMaxSize
);
```

**Description**

Sets the max size of recent file list item.

**Parameter**


---

lpsRecentFileName	[In] The input custom recent file list name.
-------------------	--

---

nMaxSize	[In] The input max size of recent file list item.
----------	---

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1394

**Related method**

[FRAppCreateCustomRecentFileList](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.3](#)

**FRAppSetEnableJS****Syntax**

```
void FRAppSetEnableJS (
    FS\_BOOL bEnable
);
```

**Description**

Sets the javascript to be enabled or not.

**Parameter**


---

bEnable	[In] Sets the javascript to be enabled or not.
---------	--

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1969

**Since**

[SDK LATEEST VERSION > 8.1](#)

**FRAppSetMainframeShow****Syntax**

```
void FRAppSetMainframeShow (
    FS\_BOOL bShow
);
```

**Description**

Sets whether the main frame will be shown or not when Foxit Reader starts up.

**Parameter**

---

bShow	[In] Whether the main frame will be shown or not when Foxit Reader starts up.
-------	---

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1904

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRApplSetMetadataOption****Syntax**

```
void FRApplSetMetadataOption (
    FS\_BOOL bCompress
);
```

**Description**

Sets the option to the document to control whether the metadata is to be compressed or not.

**Parameter**

---

bCompress	[In] The input option value, true: compress, false: uncompress.
-----------	---

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 1708

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRApplSetOwnUndoMode**

**Syntax**

```
void FRAppSetOwnUndoMode (
    FR\_Document frDoc,
    FS\_BOOL bOwnUndoMode
);
```

**Description**

Sets the own undo mode.

**Parameter**


---

frDoc	[In] The input document.
bOwnUndoMode	[In] <a href="#">TRUE</a> if you want to sets the own undo mode, otherwise not.

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 1318

**Related method**

[FRAppAddUndoRedoItem](#)  
[FRAppUndoRedoIsEditing](#)  
[FRAppUndoRedoBeginEdit](#)  
[FRAppUndoRedoEndEdit](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRAppSetRecentFileListImageByExt****Syntax**

```
FS_BOOL FRAppSetRecentFileListImageByExt (
    FS\_LPCWSTR lpwsFileExt,
    FS\_DIBitmap fileImage
);
```

**Description**

Sets the image of the file in the recent file list.

**Parameter**


---

lpwsFileExt	[In] The input file extension name, of which you want to change the image in the recent file list.
-------------	--

---

---

fileImage	[In] The input image.
-----------	-----------------------

---

**Return**

True for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1561

**Since**

[SDK\\_LATEEST\\_VERSION > 3.0](#)

**FRAppSetSubscriptionProvider****Syntax**

```
void* FRAppSetSubscriptionProvider (
    FR SubscriptionProviderCallbacks callbacks
);
```

**Description**

Registers a callbacks set for subscription provider.

**Parameter**

---

callbacks	[In] The input callbacks for subscription provider.
-----------	---

---

**Return**

The pointer to subscription provider can be destroyed by [FRAppDestroySubscriptionProvider](#).

**Head file reference**

fr\_appTempl.h: 1719

**Related method**

[FRAppDestroySubscriptionProvider](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRAppSetToolbarLocked****Syntax**

```
void FRAppSetToolbarLocked (
    FS\_BOOL bLock,
    FS\_BOOL bRecalcLayout
);
```

**Description**

Sets the toolbar to be locked or not.

#### Parameter

---

bLock	[In] <a href="#">TRUE</a> means the toolbar will be locked, otherwise not.
-------	--

---

bRecalcLayout	[In] <a href="#">TRUE</a> means the layout of toolbar will be recalced, otherwise not. Sets it <a href="#">TRUE</a> as default.
---------------	---

---

#### Return

void

#### Head file reference

fr\_appTempl.h: 1220

#### Related method

[FRAppSetToolbarLocked](#)

#### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRAppShowAdvWnd

#### Syntax

```
FS_BOOL FRAppShowAdvWnd (
    FS_INT32 nAdvWidth
);
```

#### Description

Shows the advertisement window or not.

#### Parameter

---

nAdvWidth	[In] The width of the advertisement window. Sets it as 0 to hide the advertisement window.
-----------	--

---

#### Return

TRUE for success, otherwise failure.

#### Head file reference

fr\_appTempl.h: 1505

#### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

## FRApShowFullScreen

### Syntax

```
FS_BOOL FRApShowFullScreen (void );
```

### Description

Begins full-screen mode. In full-screen mode, all window borders, the menu bar, and the toolbar are hidden. All regions of the screen outside of the page view boundary are painted by specified color. [FRApShowFullScreen](#) () is ignored if the application is already in full-screen mode, or if there are no currently open documents.

### Return

[TRUE](#) if the application enters full-screen mode; [FALSE](#) if it is already in full-screen mode or the user clicks the cancel button from dialog box describing how to exit the full-screen mode.

### Head file reference

fr\_appTempl.h: 391

### Related method

[FRApEndFullScreen](#)

## FRApShowMenuBar

### Syntax

```
void FRApShowMenuBar (  
    FS_BOOL bShow  
) ;
```

### Description

Shows/Hides menu bar.

### Parameter

---

bShow	[In] If <a href="#">TRUE</a> , the menu bar will be show, <a href="#">FALSE</a> to hide.
-------	--

---

### Return

void

### Head file reference

fr\_appTempl.h: 178

## FRApShowPreferenceDlg

### Syntax

```
FS_INT32 FRApShowPreferenceDlg (  
    FS_LPCWSTR lpwsTabTitle  
) ;
```

**Description**

Shows the preference dialog.

**Parameter**

---

lpwsTabTitle	[In]
--------------	------

---

**Return**

The result when user clicks ok button or cancel button.

**Head file reference**

fr\_appTempl.h: 266

**Related method**

[FRAppRegisterPreferencePageHandler](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRAppShowTaskPane****Syntax**

```
void FRAppShowTaskPane (
    FS_LPSTR nameOfPaneView,
    FS_BOOL bShow
);
```

**Description**

Shows or hidden a task panel view.

**Parameter**

---

nameOfPaneView	[In] The name of the pane view which would be shown or hidden.
----------------	--

---

bShow	[In] A flag indicates whether the pane view specified by <i>nameOfPaneView</i> should be shown or hidden.
-------	---

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 775

**Related method**

[FRAppRegisterTaskPaneView](#)

**Since**  
[SDK LATEEST VERSION > 1.0](#)

## FRApptoRedoBeginEdit

### Syntax

```
void FRApptoRedoBeginEdit (
    FR\_Document frDoc,
    FRApptoRedoExitEditProc pExitEditProc,
    void* clientData
);
```

### Description

When you begin to edit detail content, such as text, and you don't want to save all the editing operation to save memory, you can invoke this interface. We call it the editing mode undo-redo control.

### Parameter

frDoc	[In] The specified document.
pExitEditProc	[In] The callback will be invoked when you exit the editing mode. Then you can release the data.
clientData	[In] The client data passed to the exiting callback.

### Return

void

### Head file reference

fr\_appTempl.h: 507

### Related method

[FRApptoAddUndoRedoItem](#)  
[FRApptoUndoRedoEndEdit](#)

### Since

[SDK LATEEST VERSION > 1.0](#)

## FRApptoRedoEndEdit

### Syntax

```
void FRApptoRedoEndEdit (
    FR\_Document frDoc
);
```

**Description**

When you exit the editing mode, invoke this interface to exit the editing mode undo-redo control.

**Parameter**

---

frDoc	[In] The specified document.
-------	------------------------------

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 508

**Related method**

[FRAppAddUndoRedoItem](#)

[FRAppUndoRedoBeginEdit](#)

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRAppUndoRedoIsEditing****Syntax**

```
FS_BOOL FRAppUndoRedoIsEditing (
    FR\_Document frDoc
);
```

**Description**

Checks whether the document is being edited.

**Parameter**

---

frDoc	[In] The specified document.
-------	------------------------------

---

**Return**

[TRUE](#) means the document is being edited, otherwise not.

**Head file reference**

fr\_appTempl.h: 506

**Related method**

[FRAppAddUndoRedoItem](#)

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRAppUnRegisterCaptureHandler****Syntax**

```
FS_BOOL FRAppUnRegisterCaptureHandler (
    FR\_CaptureCallbacks callbacks
);
```

**Description**

Unregisters the capture handler and releases the memory.

**Parameter**


---

callbacks	[In] A structure containing the capture handler's callback functions.
-----------	---

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_appTempl.h: 570

**FRAppUnRegisterCmdMsgEventHandler****Syntax**

```
FS_BOOL FRAppUnRegisterCmdMsgEventHandler (
    void* cmdMsgEventHandler
);
```

**Description**

Unregisters the input cmd msg event.

**Parameter**


---

cmdMsgEventHandler	[In] The input cmd msg event handler returned by <a href="#">FRAppRegisterCmdMsgEventHandler</a> .
--------------------	--

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1240

**Related method**

[FRAppRegisterCmdMsgEventHandler](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

## FRAppUnRegisterMousePtHandler

### Syntax

```
FS_BOOL FRAppUnRegisterMousePtHandler (
    FR\_MousePtCallbacks callbacks
);
```

### Description

Unregisters the mouse point handler and releases the memory.

### Parameter

---

callbacks	[In] A structure containing the mouse point handler's callback functions.
-----------	---

---

### Return

[TRUE](#) means successful, otherwise not.

### Head file reference

fr\_appTempl.h: 1093

## FRAppUnRegisterSecurityHandler

### Syntax

```
void FRAppUnRegisterSecurityHandler (
    FS\_LPCSTR name
);
```

### Description

Unregisters a security handler.

### Parameter

---

name	[In] The name of the security handler.
------	--

---

### Return

void

### Head file reference

fr\_appTempl.h: 448

### Related method

[FRAppRegisterSecurityHandler](#)

## FRAppUnRegisterSelectionHandler

### Syntax

```
FS_BOOL FRAppUnRegisterSelectionHandler (
    FR\_SelectionCallbacks callbacks
);
```

**Description**

Unregisters the selection handler and releases the memory.

**Parameter**

---

callbacks	[In] A structure containing the selection handler's callback functions.
-----------	---

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_appTempl.h: 547

**FRAppUnRegisterWndProvider****Syntax**

```
FS_BOOL FRAppUnRegisterWndProvider (
    FS\_LPCSTR name
);
```

**Description**

Unregisters the window provider by name.

**Parameter**

---

name	[In] The input name of the window provider.
------	---

---

**Return**

[TRUE](#) means success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 1169

**Related method**

[FRAppRegisterWndProvider](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

# FR\_BulbMsgCenter

## Description

### Functions

#### Functions summary

**FRBulbMsgCenterAddMessage**

Adds a bulb message to the bulb message center.

**FRBulbMsgCenterShowMessage**

Shows the specified bulb message or not.

**FRBulbMsgCenterSetCheck**

Whether to check the check box or not.

**FRBulbMsgCenterSetOpBtnEnable**

Whether to enable the specified operation button or not.

**FRBulbMsgCenterIsMessageExist**

Checks whether the bulb message exists or not.

**FRBulbMsgCenterIsOpBtnEnable**

Checks whether the operation button is enabled or not.

**FRBulbMsgCenterSetOpBtnTitle**

Sets the title of the operation button.

**FRBulbMsgCenterSetMessageContent**

Sets the content of the bulb message.

**FRBulbMsgCenterGetButtonRect**

Gets the rectangle of the specified operation button.

**FRBulbMsgCenterAddMessage2**

Adds a bulb message to the bulb message center.

**FRBulbMsgCenterShowMessage2**

Shows the specified bulb message or not.

**FRBulbMsgCenterAddMessage3**

Adds a bulb message to the bulb message center.

**FRBulbMsgCenterAddMessage4**

Adds a bulb message to the bulb message center.

#### Functions detail

##### FRBulbMsgCenterAddMessage

###### Syntax

```
FS_BOOL FRBulbMsgCenterAddMessage (
    FR\_Document frDoc,
    pMsgInfo
);
```

###### Description

Adds a bulb message to the bulb message center.

###### Parameter

---

frDoc	[In] The associated document of the bulb message.
-------	---

---

---

pMsgInfo	[In] The input bulb message information.
----------	--

---

**Return**

TRUE for success, otherwise not.

**Head file reference**

fr\_barTempl.h: 6193

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FRBulbMsgCenterShowMessage****Syntax**

```
void FRBulbMsgCenterShowMessage (
    FR\_Document frDoc,
    FS\_LPCSTR lpsMsgName,
    FS\_BOOL bShow
);
```

**Description**

Shows the specified bulb message or not.

**Parameter**


---

frDoc	[In] The associated document of the bulb message.
-------	---

---

lpsMsgName	[In] The specified name of the bulb message.
------------	--

---

bShow	[In] Whether to show the bulb message or not.
-------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6205

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FRBulbMsgCenterSetCheck**

**Syntax**

```
void FRBulbMsgCenterSetCheck (
    FR\_Document frDoc,
    FS\_LPCSTR lpsMsgName,
    FS\_BOOL bCheck
);
```

**Description**

Whether to check the check box or not.

**Parameter**

---

frDoc	[In] The associated document of the bulb message.
-------	---

---

lpsMsgName	[In] The specified name of the bulb message.
------------	--

---

bCheck	[In] Whether to check the check box or not.
--------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6218

**Related method****Since**

[SDK LATEEST VERSION > 7.3](#)

[FRBulbMsgCenterSetOpBtnEnable](#)

**Syntax**

```
void FRBulbMsgCenterSetOpBtnEnable (
    FR\_Document frDoc,
    FS\_LPCSTR lpsMsgName,
    FS\_LPCSTR opBtnName,
    FS\_BOOL bEnable
);
```

**Description**

Whether to enable the specified operation button or not.

**Parameter**

---

frDoc	[In] The associated document of the bulb message.
-------	---

---

---

lpsMsgName	[In] The specified name of the bulb message.
opBtnName	[In] The specified name of the operation button.
bEnable	[In] Whether to enable the specified operation button or not.

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6231

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.3](#)**FRBulbMsgCenterIsMessageExist****Syntax**

```
FS_BOOL FRBulbMsgCenterIsMessageExist (
    FR\_Document frDoc,
    FS\_LPCSTR lpsMsgName
);
```

**Description**

Checks whether the bulb message exists or not.

**Parameter**


---

frDoc	[In] The associated document of the bulb message.
lpsMsgName	[In] The specified name of the bulb message.

---

**Return**

TRUE if the bulb message exists, otherwise FALSE.

**Head file reference**

fr\_barTempl.h: 6245

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FRBulbMsgCenterIsOpBtnEnable****Syntax**

```
FS_BOOL FRBulbMsgCenterIsOpBtnEnable (
    FR\_Document frDoc,
    FS\_LPCSTR lpsMsgName,
    FS\_LPCSTR opBtnName
);
```

**Description**

Checks whether the operation button is enabled or not.

**Parameter**

frDoc	[In] The associated document of the bulb message.
lpsMsgName	[In] The specified name of the bulb message.
opBtnName	[In] The specified name of the operation button.

**Return**

TRUE if the operation button is enabled, otherwise FALSE.

**Head file reference**

fr\_barTempl.h: 6257

**Related method****Since**

[SDK LATEST VERSION > 7.3](#)

**FRBulbMsgCenterSetOpBtnTitle****Syntax**

```
void FRBulbMsgCenterSetOpBtnTitle (
    FR\_Document frDoc,
    FS\_LPCSTR lpsMsgName,
    FS\_LPCSTR opBtnName,
    FS\_LPCWSTR opBtnTitle
);
```

**Description**

Sets the title of the operation button.

**Parameter**

frDoc	[In] The associated document of the bulb message.
-------	---

---

lpsMsgName	[In] The specified name of the bulb message.
------------	--

---

opBtnName	[In] The specified name of the operation button.
-----------	--

---

opBtnTitle	[In] The input title of the operation button.
------------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6270

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.3.1](#)**FRBulbMsgCenterSetMessageContent****Syntax**

```
void FRBulbMsgCenterSetMessageContent (
    FR\_Document frDoc,
    FS\_LPCSTR lpsMsgName,
    FS\_LPCWSTR lpwsMsgContent
);
```

**Description**

Sets the content of the bulb message.

**Parameter**


---

frDoc	[In] The associated document of the bulb message.
-------	---

---

lpsMsgName	[In] The specified name of the bulb message.
------------	--

---

lpwsMsgContent	[In] The input content of the bulb message.
----------------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6284

**Related method****Since**

[SDK LATEST VERSION > 7.3.1](#)**FRBulbMsgCenterGetButtonRect****Syntax**

```
void FRBulbMsgCenterGetButtonRect (
    FR\_Document frDoc,
    FS\_LPCSTR lpsMsgName,
    FS\_LPCSTR opBtnName,
    FS\_Rect* outRect
);
```

**Description**

Gets the rectangle of the specified operation button.

**Parameter**

frDoc	[In] The associated document of the bulb message.
lpsMsgName	[In] The specified name of the bulb message.
opBtnName	[In] The specified name of the operation button.
outRect	[Out] It receives the rectangle of the specified operation button.

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6297

**Related method****Since**[SDK LATEST VERSION > 7.3.1](#)**FRBulbMsgCenterAddMessage2****Syntax**

```
FS_BOOL FRBulbMsgCenterAddMessage2 (
    HWND hView,
    pMsgInfo
);
```

**Description**

Adds a bulb message to the bulb message center.

**Parameter**


---

hView	[In] The associated window of the bulb message.
pMsgInfo	[In] The input bulb message information.

---

**Return**

TRUE for success, otherwise not.

**Head file reference**

fr\_barTempl.h: 6311

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRBulbMsgCenterShowMessage2****Syntax**

```
void FRBulbMsgCenterShowMessage2 (
    HWND hView,
    FS LPCSTR lpsMsgName,
    FS BOOL bShow
);
```

**Description**

Shows the specified bulb message or not.

**Parameter**


---

hView	[In] The associated window of the bulb message.
lpsMsgName	[In] The specified name of the bulb message.
bShow	[In] Whether to show the bulb message or not.

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6323

**Related method**

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

## FRBulbMsgCenterAddMessage3

**Syntax**

```
FS_BOOL FRBulbMsgCenterAddMessage3 (
    FR\_Document frDoc,
    FRBULBMESSAGEINFO2 msgInfo2
);
```

**Description**

Adds a bulb message to the bulb message center.

**Parameter**

---

frDoc	[In] The associated document of the bulb message.
-------	---

---

msgInfo2	[In] The input bulb message information.
----------	--

---

**Return**

TRUE for success, otherwise not.

**Head file reference**

fr\_barTempl.h: 6336

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 8.0.2](#)

## FRBulbMsgCenterAddMessage4

**Syntax**

```
FS_BOOL FRBulbMsgCenterAddMessage4 (
    HWND hView,
    FRBULBMESSAGEINFO2 msgInfo2
);
```

**Description**

Adds a bulb message to the bulb message center.

**Parameter**

---

hView	[In] The associated window of the bulb message.
-------	---

---

---

msgInfo2	[In] The input bulb message information.
----------	--

---

**Return**

TRUE for success, otherwise not.

**Head file reference**

fr\_barTempl.h: 6348

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 8.0.2](#)

## FR\_CloudLoginProvider

**[Return from](#)** **[Used by](#)**

### Description

#### Returned from

[FRCloudLoginProviderGet](#)  
[FRCloudLoginProviderSet](#)

#### Used by

[FRCloudLoginProviderDestroy](#)  
[FRCloudLoginProviderGetUserInfo](#)  
[FRCloudLoginProviderIsLogIn](#)  
[FRCloudLoginProviderSet](#)  
[FRCloudLoginProviderSignIn](#)  
[FRCloudLoginProviderSignOut](#)

### Callbacks

#### Callbacks summary

[FRCloudLoginProviderOnIsLogIn](#)

A callback for the service provider of cloud login.

[FRCloudLoginProviderOnSignIn](#)

A callback for the service provider of cloud login.

[FRCloudLoginProviderOnSignOut](#)

A callback for the service provider of cloud login.

[FRCloudLoginProviderOnGetUserInfo](#)

A callback for the service provider of cloud login.

#### Callbacks detail

[FRCloudLoginProviderOnIsLogIn](#)

**Syntax**

```
typedef FS_BOOL (*FRCloudLoginProviderOnIsLogIn)(
    fwsHost,
    FS\_LPVOID clientData
);
```

**Description**

A callback for the service provider of cloud login. It is called to check whether the user has logged in or not.

**Parameter**

fwsHost	[In] The Host url.
---------	--------------------

clientData	[In] The user-supplied data.
------------	------------------------------

**Return**

TRUE indicates that the user has logged in, otherwise not.

**Head file reference**

fr\_appExpT.h: 6728

**Group**

[FR\\_CloudLoginProviderCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRCloudLoginProviderOnSignIn****Syntax**

```
typedef FS_BOOL (*FRCloudLoginProviderOnSignIn)(
    fwsHost,
    FS\_LPVOID clientData
);
```

**Description**

A callback for the service provider of cloud login. It is called to sign into the cloud.

**Parameter**

fwsHost	[In] The Host url.
---------	--------------------

clientData	[In] The user-supplied data.
------------	------------------------------

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_appExpT.h: 6743

**Group**[FR\\_CloudLoginProviderCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.3.1](#)**FRCLOUDLOGINPROVIDERONSIGNOUT****Syntax**

```
typedef FS_BOOL (*FRCLOUDLOGINPROVIDERONSIGNOUT)(  
    fwsDomain,  
    FS\_LPVOID clientData  
);
```

**Description**

A callback for the service provider of cloud login. It is called to sign out the cloud.

**Parameter**

---

fwsDomain	[In] The Domain url.
-----------	----------------------

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_appExpT.h: 6758

**Group**[FR\\_CloudLoginProviderCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.3.1](#)**FRCLOUDLOGINPROVIDERONGETUSERINFO****Syntax**

```
typedef FS_BOOL (*FRCLOUDLOGINPROVIDERONGETUSERINFO)(  
    fwsDomain,  
    FS\_LPVOID clientData,  
    FR\_Login\_UserInfo* pUserInfo  
);
```

**Description**

A callback for the service provider of cloud login. It is called to get the user information.

**Parameter**


---

fwsDomain	[In] The Domain url.
clientData	[In] The user-supplied data.
pUserInfo	[Out] It receives the user information.

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_appExpT.h: 6774

**Group**

[FR\\_CloudLoginProviderCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 7.3.1](#)

## Structures

### Structures summary

#### [FR\\_Login\\_UserInfo](#)

A data structure representing the user information of cloud login.

### Structs detail

#### FR\_Login\_UserInfo

**Syntax**

```
typedef struct __FR_Login_UserInfo__{
    FS_WideString wsUserID,
    FS_WideString wsEmail,
    FS_WideString wsToken,
    FS_WideString wsFirstName,
    FS_WideString wsLastName,
    wsUserID = NULL,
    wsEmail = NULL,
    wsToken = NULL,
    wsFirstName = NULL,
    wsLastName = NULL
}FR_Login_UserInfo;
```

**Description**

A data structure representing the user information of cloud login.

**Head file reference**

fr\_appExpT.h: 6677

**wsUserID**

The user ID. Invokes [FSWideStringNew](#) to initialize.

**wsEmail**

The user email. Invokes [FSWideStringNew](#) to initialize.

**wsToken**

The user token. Invokes [FSWideStringNew](#) to initialize.

**wsFirstName**

The user first name. Invokes [FSWideStringNew](#) to initialize.

**wsLastName**

The user last name. Invokes [FSWideStringNew](#) to initialize.

**= NULL**

The user last name. Invokes [FSWideStringNew](#) to initialize.

**= NULL**

The user last name. Invokes [FSWideStringNew](#) to initialize.

**= NULL**

The user last name. Invokes [FSWideStringNew](#) to initialize.

**= NULL**

The user last name. Invokes [FSWideStringNew](#) to initialize.

## Functions

### Functions summary

**[FRCloudLoginProviderDestroy](#)**

Destroys the service provider of cloud login.

**[FRCloudLoginProviderGet](#)**

Gets the service provider of cloud login.

**[FRCloudLoginProviderGetUserInfo](#)**

Gets the user information.

**[FRCloudLoginProviderIsLogIn](#)**

Checks whether the user has logged in or not.

**[FRCloudLoginProviderSet](#)**

Sets the service provider of cloud login.

**[FRCloudLoginProviderSignIn](#)**

Signs into the cloud.

**[FRCloudLoginProviderSignOut](#)**

Signs out the cloud.

## Functions detail

### [FRCloudLoginProviderDestroy](#)

**Syntax**

```
void FRCloudLoginProviderDestroy (
    FR\_CloudLoginProvider loginProvider
);
```

**Description**

Destroys the service provider of cloud login.

**Parameter**

---

loginProvider	[In] The input service provider of cloud login.
---------------	---

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 4185

**Related method**

**[FRCloudLoginProviderSet](#)**

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

### [FRCloudLoginProviderGet](#)

**Syntax**

```
FR\_CloudLoginProvider FRCloudLoginProviderGet (void );
```

**Description**

Gets the service provider of cloud login.

**Return**

The service provider of cloud login.

**Head file reference**

fr\_appTempl.h: 4204

**Related method**[FRCLOUDLOGINPROVIDERSET](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.3.1](#)**FRCLOUDLOGINPROVIDERGETUSERINFO****Syntax**

```
FS_BOOL FRCLOUDLOGINPROVIDERGETUSERINFO (
    FR_CLOUDLOGINPROVIDER loginProvider,
    FR_LOGIN_USERINFO* pUserInfo
);
```

**Description**

Gets the user information.

**Parameter**

loginProvider	[In] The input service provider of cloud login.
---------------	---

pUserInfo	[Out] It receives the user information.
-----------	---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 4247

**Related method**[FRCLOUDLOGINPROVIDERSET](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.3.1](#)**FRCLOUDLOGINPROVIDERISLOGIN****Syntax**

```
FS_BOOL FRCLOUDLOGINPROVIDERISLOGIN (
    FR_CLOUDLOGINPROVIDER loginProvider
);
```

**Description**

Checks whether the user has logged in or not.

**Parameter**

---

loginProvider	[In] The input service provider of cloud login.
---------------	---

---

**Return**

TRUE indicates that the user has logged in, otherwise not.

**Head file reference**

fr\_appTempl.h: 4214

**Related method**

[FRCLOUDLOGINPROVIDERSET](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRCLOUDLOGINPROVIDERSET****Syntax**

```
FR_CloudLoginProvider FRCLOUDLOGINPROVIDERSET (
    FR\_CloudLoginProviderCallbacks callbacks
);
```

**Description**

Sets the service provider of cloud login.

**Parameter**


---

callbacks	[In] The callback set for the service provider of cloud login.
-----------	--

---

**Return**

The [\\*FR\\_CloudLoginProvider](#) object represents the service provider of cloud login. Destroys it by invoking [FRCLOUDLOGINPROVIDERDESTROY](#).

**Head file reference**

fr\_appTempl.h: 4182

**Related method**

[FRCLOUDLOGINPROVIDERDESTROY](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRCLOUDLOGINPROVIDERSIGNIN****Syntax**

```
FS_BOOL FRCLOUDLOGINPROVIDERSIGNIN (
    FR\_CloudLoginProvider loginProvider
);
```

**Description**

Signs into the cloud.

**Parameter**

---

loginProvider	[In] The input service provider of cloud login.
---------------	---

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 4225

**Related method**

[FRCloudLoginProviderSet](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

## FRCloudLoginProviderSignOut

**Syntax**

```
FS_BOOL FRCloudLoginProviderSignOut (
    FR\_CloudLoginProvider loginProvider
);
```

**Description**

Signs out the cloud.

**Parameter**

---

loginProvider	[In] The input service provider of cloud login.
---------------	---

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 4236

**Related method**

[FRCloudLoginProviderSet](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

# FR\_Cursor

## [Return from Used by](#)

### Description

A data structure representing the cursor.

### Returned from

[FRSysGetCursor](#)  
[FRSysLoadStandarCursor](#)

### Used by

[FRSysSetCursor](#)

## Definitions

### Definitions summary

#### [FR\\_CURSOR\\_TYPE\\_ARROW](#)

Standard arrow cursor See [FRSysLoadStandarCursor](#).

#### [FR\\_CURSOR\\_TYPE\\_CROSS](#)

Cross-hair cursor for selection

#### [FR\\_CURSOR\\_TYPE\\_HELP](#)

Help cursor

#### [FR\\_CURSOR\\_TYPE\\_SIZEALL](#)

A four-pointed arrow. The cursor to use to resize a window.

#### [FR\\_CURSOR\\_TYPE\\_SIZENESW](#)

Two-headed arrow with ends at upper right and lower left.

#### [FR\\_CURSOR\\_TYPE\\_SIZENS](#)

Vertical two-headed arrow.

#### [FR\\_CURSOR\\_TYPE\\_SIZENWSE](#)

Two-headed arrow with ends at upper left and lower right.

#### [FR\\_CURSOR\\_TYPE\\_SIZEWE](#)

Horizontal two-headed arrow.

#### [FR\\_CURSOR\\_TYPE\\_UPARROW](#)

Arrow that points straight up.

#### [FR\\_CURSOR\\_TYPE\\_WAIT](#)

Hourglass cursor used when Windows performs a time-consuming task.

### Definitions detail

#### [FR\\_CURSOR\\_TYPE\\_ARROW](#)

##### Syntax

```
#define FR_CURSOR_TYPE_ARROW 0
```

##### Description

Standard arrow cursor See [FRSysLoadStandarCursor](#) .

**Group**

[FRCursorTypeFlags](#)

**Head file reference**

fr\_sysExpT.h: 110

## FR\_CURSOR\_TYPE\_CROSS

**Syntax**

```
#define FR_CURSOR_TYPE_CROSS 1
```

**Description**

Cross-hair cursor for selection

**Group**

[FRCursorTypeFlags](#)

**Head file reference**

fr\_sysExpT.h: 113

## FR\_CURSOR\_TYPE\_HELP

**Syntax**

```
#define FR_CURSOR_TYPE_HELP 2
```

**Description**

Help cursor

**Group**

[FRCursorTypeFlags](#)

**Head file reference**

fr\_sysExpT.h: 116

## FR\_CURSOR\_TYPE\_SIZEALL

**Syntax**

```
#define FR_CURSOR_TYPE_SIZEALL 3
```

**Description**

A four-pointed arrow. The cursor to use to resize a window.

**Group**

[FRCursorTypeFlags](#)

**Head file reference**

fr\_sysExpT.h: 119

## FR\_CURSOR\_TYPE\_SIZENESW

**Syntax**

```
#define FR_CURSOR_TYPE_SIZENESW 4
```

**Description**

Two-headed arrow with ends at upper right and lower left.

**Group**

[FRCursorTypeFlags](#)

**Head file reference**

fr\_sysExpT.h: 122

## FR\_CURSOR\_TYPE\_SIZENS

**Syntax**

```
#define FR_CURSOR_TYPE_SIZENS 5
```

**Description**

Vertical two-headed arrow.

**Group**

[FRCursorTypeFlags](#)

**Head file reference**

fr\_sysExpT.h: 125

## FR\_CURSOR\_TYPE\_SIZENWSE

**Syntax**

```
#define FR_CURSOR_TYPE_SIZENWSE 6
```

**Description**

Two-headed arrow with ends at upper left and lower right.

**Group**

[FRCursorTypeFlags](#)

**Head file reference**

fr\_sysExpT.h: 128

## FR\_CURSOR\_TYPE\_SIZEWE

**Syntax**

```
#define FR_CURSOR_TYPE_SIZEWE 7
```

**Description**

Horizontal two-headed arrow.

**Group**

[FRCursorTypeFlags](#)

**Head file reference**

fr\_sysExpT.h: 131

## FR\_CURSOR\_TYPE\_UPARROW

**Syntax**

```
#define FR_CURSOR_TYPE_UPARROW 8
```

**Description**

Arrow that points straight up.

**Group**

[FRCursorTypeFlags](#)

**Head file reference**

fr\_sysExpT.h: 134

## FR\_CURSOR\_TYPE\_WAIT

**Syntax**

```
#define FR_CURSOR_TYPE_WAIT 9
```

**Description**

Hourglass cursor used when Windows performs a time-consuming task.

**Group**

[FRCursorTypeFlags](#)

**Head file reference**

fr\_sysExpT.h: 137

## FR\_CustomSignature

**Description**

## Functions

### Functions summary

#### [\*\*FRCustomSignatureGenerateSignInfo\*\*](#)

Signs a PDF document with the [FR\\_SignatureInfo](#) and [FR\\_SignaturePosInfo](#). You have to register the signature handler by [FRCustomSignatureRegisterSignatureHandler](#) to sign the data, otherwise the data will be signed with the default standard method.

#### [\*\*FRCustomSignatureGetDefaultServer\*\*](#)

Gets the default timestamp server.

#### [\*\*FRCustomSignatureCreateSignatureHandler\*\*](#)

Creates the signature handler. Registers it by [FRCustomSignatureRegisterSignatureHandler](#). Destroys it by [FRCustomSignatureDestroySignatureHandler](#).

#### [\*\*FRCustomSignatureRegisterSignatureHandler\*\*](#)

Registers the signature handler. You can customize the process signing the data and the process verifying the digital signature.

#### [\*\*FRCustomSignatureDestroySignatureHandler\*\*](#)

Destroys the signature handler returned by [FRCustomSignatureCreateSignatureHandler](#).

#### [\*\*FRCustomSignatureSetSignatureVerify\*\*](#)

Verifies the specified signature.

#### [\*\*FRCustomSignatureGetDocSignatureCount\*\*](#)

Gets the signature count.

#### [\*\*FRCustomSignatureGetSignatureBaseInfo\*\*](#)

Gets the specified base info of the signature.

#### [\*\*FRCustomSignatureClearSignature\*\*](#)

Clears the specified signature.

### Functions detail

#### FRCustomSignatureGenerateSignInfo

##### Syntax

```
FS_BOOL FRCustomSignatureGenerateSignInfo (
    FR_SignatureInfo* pSgInfo,
    FR_SignaturePosInfo* pSgPosInfo
);
```

##### Description

Signs a PDF document with the [FR\\_SignatureInfo](#) and [FR\\_SignaturePosInfo](#). You have to register the signature handler by [FRCustomSignatureRegisterSignatureHandler](#) to sign the data, otherwise the data will be signed with the default standard method.

##### Parameter

pSgInfo	[In] To fill the signature info.
---------	----------------------------------

pSgPosInfo	[In] To fill the signature position info.
------------	---

##### Return

TRUE for success, otherwise failure.

**Head file reference**

fr\_docTempl.h: 1278

**Since**

[SDK LATEST VERSION > 7.2.2](#)

**FRCustomSignatureGetDefaultServer****Syntax**

```
FS_BOOL FRCustomSignatureGetDefaultServer (
    FR\_SignatureTimestampServer* pSgTMServer
);
```

**Description**

Gets the default timestamp server.

**Parameter**

---

pSgTMServer	[Out] It receives the default timestamp server.
-------------	---

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_docTempl.h: 1290

**Since**

[SDK LATEST VERSION > 7.2.2](#)

**FRCustomSignatureCreateSignatureHandler****Syntax**

```
void* FRCustomSignatureCreateSignatureHandler (
    FR\_SignatureHandlerCallbacks callbacks
);
```

**Description**

Creates the signature handler. Registers it by [FRCustomSignatureRegisterSignatureHandler](#). Destroys it by [FRCustomSignatureDestroySignatureHandler](#).

**Parameter**

---

callbacks	[In] The callback set for signature handler.
-----------	--

---

**Return**

The pointer to the signature handler.

**Head file reference**

fr\_docTempl.h: 1300

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

## FRCustomSignatureRegisterSignatureHandler

**Syntax**

```
FS_BOOL FRCustomSignatureRegisterSignatureHandler (
    void* pSignatureHandler
);
```

**Description**

Registers the signature handler. You can customize the process signing the data and the process verifying the digital signature.

**Parameter**

---

pSignatureHandler	[In] The input pointer to signature handler.
-------------------	--

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_docTempl.h: 1280

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

## FRCustomSignatureDestroySignatureHandler

**Syntax**

```
void FRCustomSignatureDestroySignatureHandler (
    void* pSignatureHandler
);
```

**Description**

Destroys the signature handler returned by [FRCustomSignatureCreateSignatureHandler](#).

**Parameter**

---

pSignatureHandler	[In] The input pointer to signature handler.
-------------------	--

---

**Return**

void.

**Head file reference**

fr\_docTempl.h: 1301

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRCustomSignatureSetSignatureVerify****Syntax**

```
FS_BOOL FRCustomSignatureSetSignatureVerify (
    FR\_Document frDoc,
    const unsigned char* pSignedData,
    unsigned long ulSignedDataLen
);
```

**Description**

Verifies the specified signature.

**Parameter**

---

frDoc

[In] The input document.

---

pSignedData

[In] The signed data.

---

ulSignedDataLen

[In] The length of the signed data.

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_docTempl.h: 1330

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FRCustomSignatureGetDocSignatureCount****Syntax**

```
FS_INT32 FRCustomSignatureGetDocSignatureCount (
    FR\_Document frDoc
);
```

**Description**

Gets the signature count.

**Parameter**

---

frDoc	[In] The input document.
-------	--------------------------

---

**Return**

The signature count.

**Head file reference**

fr\_docTempl.h: 1342

**Since**

[SDK LATEEST VERSION > 8.0](#)

**FRCustomSignatureGetSignatureBaseInfo****Syntax**

```
FS_BOOL FRCustomSignatureGetSignatureBaseInfo (
    FR\_Document frDoc,
    FS\_INT32 nIndex,
    FR\_SignatureBaseInfo* pInfo
);
```

**Description**

Gets the specified base info of the signature.

**Parameter**

---

frDoc	[In] The input document.
-------	--------------------------

---

---

nIndex	[In] The index of the signature.
--------	----------------------------------

---

---

pInfo	[Out] It receives the base info of the signature.
-------	---

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_docTempl.h: 1352

**Since**

[SDK LATEEST VERSION > 8.0](#)

**FRCustomSignatureClearSignature**

**Syntax**

```
FS_BOOL FRCustomSignatureClearSignature (
    FR\_Document frDoc,
    FS\_INT32 nIndex
);
```

**Description**

Clears the specified signature.

**Parameter**

frDoc	[In] The input document.
nIndex	[In] The index of the signature.

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_docTempl.h: 1364

**Since**

[SDK\\_LATEEST\\_VERSION > 8.0](#)

## FR\_Document

### [Return from Used by](#)

#### Description

A [FR\\_Document](#) is a view of a PDF document in a window.

Usually there is one [FR\\_Document](#) per displayed document, but some times(such as New Window), there may be two or more [FR\\_Document](#) per displayed document. Unlike a [FPD\\_Document](#), an [FR\\_Document](#) has a window associated with it. And a [FPD\\_Document](#) y have one or more associated [FR\\_Document](#).

#### Returned from

[FRAppCreateBlankDoc](#)  
[FRAppGetActiveDocOfPDDoc](#)  
[FRAppGetDocOfPDDoc](#)  
[FRDocViewGetDocument](#)  
[FRPageViewGetDocument](#)  
[FRDocFromPDDoc](#)  
[FRDocOpenFromFile](#)  
[FRDocOpenFromFile2](#)  
[FRDocOpenFromPDDoc](#)  
[FRDocOpenFromPDDoc2](#)

## Used by

[FRAppAddFileAttachment](#)  
[FRAppAddUndoRedoItem](#)  
[FRAppCollectDocActionData](#)  
[FRAppConvertToPDFA](#)  
[FRAppExitOwnUndoMode](#)  
[FRAppGetTaskPaneWndHandleByDoc](#)  
[FRAppSetActiveDocOfPDDoc](#)  
[FRAppSetOwnUndoMode](#)  
[FRAppUndoRedoBeginEdit](#)  
[FRAppUndoRedoEndEdit](#)  
[FRAppUndoRedoIsEditing](#)  
[FRBulbMsgCenterAddMessage](#)  
[FRBulbMsgCenterShowMessage](#)  
[FRBulbMsgCenterSetCheck](#)  
[FRBulbMsgCenterSetOpBtnEnable](#)  
[FRBulbMsgCenterIsMessageExist](#)  
[FRBulbMsgCenterIsOpBtnEnable](#)  
[FRBulbMsgCenterSetOpBtnTitle](#)  
[FRBulbMsgCenterSetMessageContent](#)  
[FRBulbMsgCenterGetButtonRect](#)  
[FRBulbMsgCenterAddMessage3](#)  
[FRCustomSignatureSetSignatureVerify](#)  
[FRCustomSignatureGetDocSignatureCount](#)  
[FRCustomSignatureGetSignatureBaseInfo](#)  
[FRCustomSignatureClearSignature](#)  
[FRTextSelectToolCreate](#)  
[FRDocAddToCurrentSelection](#)  
[FRDocCanSecurityMethodBeModified](#)  
[FRDocChangeDocShowTitle](#)  
[FRDocCheckInDocumentOLE](#)  
[FRDocClearChangeMark](#)  
[FRDocClearSelection](#)  
[FRDocClose](#)  
[FRDocCopySelection](#)  
[FRDocCountDocViews](#)  
[FRDocCreateNewViewByWndProvider](#)  
[FRDocDeleteSelection](#)  
[FRDocDidDeletePages](#)  
[FRDocDidInsertPages](#)  
[FRDocDidResizePage](#)  
[FRDocDidRotatePage](#)  
[FRDocDoPassWordEncrypt](#)  
[FRDocDoPrint](#)  
[FRDocDoPrintSilently](#)  
[FRDocDoSave](#)  
[FRDocDoSave2](#)  
[FRDocDoSaveAs](#)  
[FRDocDoSaveAs2](#)  
[FRDocEnableRunScript](#)  
[FRDocForbidChangeMark](#)  
[FRDocGenerateRedactions](#)  
[FRDocGenerateUR3Permission](#)  
[FRDocGetAppPermissions](#)  
[FRDocGetAppPermissionsII](#)

[FRDocGetChangeMark](#)  
[FRDocGetCreateDocSource](#)  
[FRDocGetCreateDocSourceFileName](#)  
[FRDocGetCreateDocSourceFilePath](#)  
[FRDocGetCurCapture](#)  
[FRDocGetCurCaptureType](#)  
[FRDocGetCurrentDocView](#)  
[FRDocGetCurrentSecurityMethodName](#)  
[FRDocGetCurrentWndProvider](#)  
[FRDocGetCurSelection](#)  
[FRDocGetCurSelectionType](#)  
[FRDocGetDocFrameHandler](#)  
[FRDocGetDocShowTitle](#)  
[FRDocGetDocumentType](#)  
[FRDocGetDocView](#)  
[FRDocGetFilePath](#)  
[FRDocGetMergedPermissions](#)  
[FRDocGetOriginalType](#)  
[FRDocGetParser](#)  
[FRDocGetPDDoc](#)  
[FRDocGetPDFCreator](#)  
[FRDocGetPermissions](#)  
[FRDocGetPermissionsII](#)  
[FRDocGetReviewType](#)  
[FRDocGetSignaturePermissions](#)  
[FRDocGetTextSelectTool](#)  
[FRDocGetUIParentWnd](#)  
[FRDocGetWndProviderByName](#)  
[FRDocHasRedactAnnot](#)  
[FRDocIsEnableRunScript](#)  
[FRDocIsEncrypted](#)  
[FRDocIsImageBasedDocument](#)  
[FRDocIsInProtectedViewMode](#)  
[FRDocIsMemoryDoc](#)  
[FRDocIsShowInBrowser](#)  
[FRDocIsValidAnnot](#)  
[FRDocIsVisible](#)  
[FRDocIsWillReopen](#)  
[FRDocKillFocusAnnot](#)  
[FRDocLoadAnnots](#)  
[FRDocParsePage](#)  
[FRDocPrintPages](#)  
[FRDocPrintSetup](#)  
[FRDocReleaseCurCapture](#)  
[FRDocReloadPage](#)  
[FRDocRemoveFromSelection](#)  
[FRDocRemoveSecurityMethod](#)  
[FRDocResetDocTitleColor](#)  
[FRDocSetAppPermissions](#)  
[FRDocSetChangeMark](#)  
[FRDocSetCreateDocSource](#)  
[FRDocSetCurCapture](#)  
[FRDocSetCurrentWndProvider](#)  
[FRDocSetCurSelection](#)  
[FRDocSetCustomSecurity](#)  
[FRDocSetDocEncrypted](#)  
[FRDocSetDocTitleColor](#)

[FRDocSetDRMSecurity](#)  
[FRDocSetFocusAnnot](#)  
[FRDocSetInputPasswordProc](#)  
[FRDocSetMenuEnableByName](#)  
[FRDocSetOriginalType](#)  
[FRDocSetPermissions](#)  
[FRDocSetPropertiesFilePath](#)  
[FRDocSetPropertiesPDFVersion](#)  
[FRDocSetReviewType](#)  
[FRDocShowSaveProgressCancelButton](#)  
[FRDocShowSelection](#)  
[FRDocUpdateSecurityMethod](#)  
[FRDocWillDeletePages](#)  
[FRDocWillInsertPages](#)  
[FRDocWillResizePage](#)  
[FRDocWillRotatePage](#)

## Definitions

### Definitions summary

#### [FR\\_DOCTYPE\\_DYNAMIC\\_XFA](#)

The type dynamic xfa.

#### [FR\\_DOCTYPE\\_PDF](#)

The type is PDF.

#### [FR\\_DOCTYPE\\_STATIC\\_XFA](#)

The type is static XFA.

#### [FR\\_MENU\\_ENABLE\\_EMAIL](#)

email menu

#### [FR\\_MENU\\_ENABLE\\_SAVEAS](#)

save as menu

#### [FR\\_MENU\\_ENABLE\\_SNAPSHOT](#)

snapshot menu

#### [FR\\_MENU\\_ENABLE\\_STAMP](#)

stamp menu

### Definitions detail

#### [FR\\_DOCTYPE\\_DYNAMIC\\_XFA](#)

##### **Syntax**

```
#define FR_DOCTYPE_DYNAMIC_XFA 1
```

##### **Description**

The type dynamic xfa.

##### **Group**

[FRDocTypes](#)

##### **Head file reference**

fr\_docExpT.h: 99

## FR\_DOCTYPE\_PDF

**Syntax**

```
#define FR_DOCTYPE_PDF 0
```

**Description**

The type is PDF.

**Group**

[FRDocTypes](#)

**Head file reference**

fr\_docExpT.h: 96

## FR\_DOCTYPE\_STATIC\_XFA

**Syntax**

```
#define FR_DOCTYPE_STATIC_XFA 2
```

**Description**

The type is static XFA.

**Group**

[FRDocTypes](#)

**Head file reference**

fr\_docExpT.h: 102

## FR\_MENU\_ENABLE\_EMAIL

**Syntax**

```
#define FR_MENU_ENABLE_EMAIL "Email"
```

**Description**

email menu

**Group**

[FRMenuEnableNames](#)

**Head file reference**

fr\_docExpT.h: 79

## FR\_MENU\_ENABLE\_SAVEAS

**Syntax**

```
#define FR_MENU_ENABLE_SAVEAS "SaveAs"
```

**Description**

save as menu

**Group**

[FRMenuEnableNames](#)

**Head file reference**

fr\_docExpT.h: 76

## FR\_MENU\_ENABLE\_SNAPSHOT

**Syntax**

```
#define FR_MENU_ENABLE_SNAPSHOT "Snapshot"
```

**Description**

snapshot menu

**Group**

[FRMenuEnableNames](#)

**Head file reference**

fr\_docExpT.h: 82

## FR\_MENU\_ENABLE\_STAMP

**Syntax**

```
#define FR_MENU_ENABLE_STAMP "Stamp"
```

**Description**

stamp menu

**Group**

[FRMenuEnableNames](#)

**Head file reference**

fr\_docExpT.h: 85

## Enumerations

### Enumerations summary

**[FRCREATEDOC\\_SOURCE](#)**

The source type of the document.

**[FRORIGINALDOC\\_TYPE](#)**

The original type of the document. The real format of the opened document is PDF, but its wrapper format may be PPDF. See [FRDOCGETORIGINALTYPE](#).

## Enumerations detail

### FRCREATEDOC SOURCE

#### Syntax

```
enum FRCREATEDOC SOURCE{  
    FR_DOC_SOURCE_NORMAL,  
    FR_DOC_SOURCE_BLANK,  
    FR_DOC_SOURCE_FROM_FILE,  
    FR_DOC_SOURCE_FROM_MULTIPLE_FILES,  
    FR_DOC_SOURCE_FROM_SCANNER,  
    FR_DOC_SOURCE_FROM_CLIPBOARD,  
    FR_DOC_SOURCE_PDF_PORTFOLIO  
};
```

#### Description

The source type of the document.

#### Head file reference

fr\_docExpT.h: 123

##### **FR\_DOC\_SOURCE\_NORMAL**

Normal.

##### **FR\_DOC\_SOURCE\_BLANK**

Blank.

##### **FR\_DOC\_SOURCE\_FROM\_FILE**

From file.

##### **FR\_DOC\_SOURCE\_FROM\_MULTIPLE\_FILES**

From Multiple Files.

##### **FR\_DOC\_SOURCE\_FROM\_SCANNER**

From Scanner.

##### **FR\_DOC\_SOURCE\_FROM\_CLIPBOARD**

From Clipboard.

##### **FR\_DOC\_SOURCE\_PDF\_PORTFOLIO**

### FROriginalDocType

#### Syntax

```
enum FROriginalDocType{  
    FR_ORI_DOCTYPE_PDF,  
    FR_ORI_DOCTYPE_DYNAMIC_XFA,  
    FR_ORI_DOCTYPE_STATIC_XFA,  
    FR_ORI_DOCTYPE_FDF,  
    FR_ORI_DOCTYPE_XDP,  
    FR_ORI_DOCTYPE_XFDF,  
    FR_ORI_DOCTYPE_PPDF  
};
```

**Description**

The original type of the document. The real format of the opened document is PDF, but its wrapper format may be PPDF. See [FRDocGetOriginalType](#) .

**Head file reference**

fr\_docExpT.h: 109

**FR\_ORI\_DOCTYPE\_PDF**

The original type of the document is PDF.

**FR\_ORI\_DOCTYPE\_DYNAMIC\_XFA**

The original type of the document is dynamic XFA.

**FR\_ORI\_DOCTYPE\_STATIC\_XFA**

The original type of the document is static PDF.

**FR\_ORI\_DOCTYPE\_FDF**

The original type of the document is FDF.

**FR\_ORI\_DOCTYPE\_XDP**

The original type of the document is XDP.

**FR\_ORI\_DOCTYPE\_XFDF**

The original type of the document is XFDF.

**FR\_ORI\_DOCTYPE\_PPDF**

## Callbacks

### Callbacks summary

**[FR\\_DocSaveAsProc](#)**

Prototype of callback function invoked by *Foxit Reader* when performs *SaveAs* .

**[FR\\_DocSaveProc](#)**

Prototype of callback function invoked by *Foxit Reader* when performs *Save* .

**[FRInputPasswordProc](#)**

Prototype of callback function invoked by *Foxit Reader* if you want to set the document password through interface instead of the password input dialog.

**[FRPasswordEncryptProc](#)**

Prototype of callback function invoked by *Foxit Reader* if you want to set the document password through the password input dialog.

### Callbacks detail

**FR\_DocSaveAsProc****Syntax**

```
typedef (*FR_DocSaveAsProc)(  
    frDoc,  
    pwszFilePath,
```

```
pProcData  
);
```

**Description**

Prototype of callback function invoked by *Foxit Reader* when performs *SaveAs* .

**Parameter**

frDoc	[In] The document to be saved as.
pwszFilePath	[In] The path where the document to be saved as.
pProcData	[In] The client data.

**Return**

void

**Head file reference****Related method**

[FRDocDoSave2](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FR\_DocSaveProc****Syntax**

```
typedef void (*FR_DocSaveProc)(  
    FR\_Document frDoc,  
    FS LPCWSTR pwszFilePath,  
    void* pProcData  
);
```

**Description**

Prototype of callback function invoked by *Foxit Reader* when performs *Save* .

**Parameter**

frDoc	[In] The document to be saved as.
pwszFilePath	[In] The path where the document to be saved.
pProcData	[In] The client data.

**Return**

void

**Head file reference**

fr\_docExpT.h: 159

**Related method**

[FRDocDoSave](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

## FRInputPasswordProc

**Syntax**

```
typedef FS_BOOL (*FRInputPasswordProc)(  
    FR_Document frDoc,  
    FS_ByteString bsPassword,  
    FS_BOOL* bReset  
)
```

**Description**

Prototype of callback function invoked by *Foxit Reader* if you want to set the document password through interface instead of the password input dialog.

**Parameter**

---

frDoc	[In] The document.
-------	--------------------

---

bsPassword	[Out] It receives the password.
------------	---------------------------------

---

bReset	[Out] It indicates whether <i>Foxit Reader</i> need to reset the password if the password is incorrect.
--------	---

---

**Return**

[TRUE](#) if the plug-in takes over the procedure, otherwise not.

**Head file reference**

fr\_docExpT.h: 174

**Related method**

[FRDocSetInputPasswordProc](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.2](#)

## FRPasswordEncryptProc

### Syntax

```
typedef void (*FRPasswordEncryptProc)(  
    FR\_Document frDoc,  
    int nAction,  
    BOOL bCancel,  
    void pVariable  
)
```

### Description

Prototype of callback function invoked by *Foxit Reader* if you want to set the document password through the password input dialog.

### Parameter

---

frDoc	[In] The document.
nAction	[In] <a href="#">0</a> means user cancel protect. <a href="#">1</a> means will start protecting document and can be canceled by <a href="#">bCancel</a> parameter. <a href="#">2</a> means the document finished protecting.
bCancel	[Out] When <a href="#">nAction</a> return <a href="#">1</a> , set <a href="#">TRUE</a> to cancel the protection.
pVariable	[Out] This is a reserved parameter.

---

### Return

void.

### Head file reference

[fr\\_docExpT.h: 191](#)

### Related method

[FRDocSetInputPasswordProc](#)

### Since

[SDK\\_LATEEST\\_VERSION > 9.1](#)

## Functions

### Functions summary

#### [FRDocAddToCurrentSelection](#)

Adds the new data to the current selection data and returns the updated one.

#### [FRDocCanSecurityMethodBeModified](#)

Whether the security applied to the document can be modified or not.

**FRDocChangeDocShowTitle**

Sets the main frame title and the document tab title.

**FRDocCheckInDocumentOLE**

Checks whether the document is opened by OLE. For example, the document is embedded in *MS Office Word* .

**FRDocClearChangeMark**

Invalidates all modification.

**FRDocClearSelection**

Clears and destroys the current selection by calling the appropriate selection server's [FRSelectionLosingSelection](#) () .

**FRDocClose**

Closes the document window.

**FRDocConvertPdfToOtherFormat**

Convert Pdf document to other format document.

**FRDocCopySelection**

Copies the current selection to the clipboard, if possible. The selection is copied if the selection handler has a [FRSelectionDoCopy](#) () callback, and the selection handler's [FRSelectionCanCopy](#) () callback returns [TRUE](#) . If the selection server does not have a [FRSelectionCanCopy](#) () method, a default value of [TRUE](#) is used. The selection is copied by calling the selection handler's [FRSelectionDoCopy](#) () callback. It only raises those exceptions raised by the selection handler's [FRSelectionDoCopy](#) () and [FRSelectionCanCopy](#) () callbacks.

**FRDocCountDocViews**

Gets the number of [FR\\_DocView](#) for specified document.

**FRDocCreateNewViewByWndProvider**

Creates the new view by [FR\\_WndProviderCallbacksRec](#) .

**FRDocDeleteSelection**

Deletes the specified document's current selection, if possible. The selection is deleted if changing the selection is currently permitted, the selection handler has an [FRSelectionDoDelete](#) () callback, and the selection server's [FRSelectionCanDelete](#) () callback returns [TRUE](#) . If the selection handler does not have a [FRSelectionCanDelete](#) () callback, a default value of [TRUE](#) is used. The selection is deleted by calling the selection handler's [FRSelectionDoDelete](#) () callback. It only raises those exceptions raised by the selection handler's [FRSelectionDoDelete](#) () and [FRSelectionCanDelete](#) () callbacks.

**FRDocDidDeletePages**

The document was deleted any pages. This notification will be broadcast to all plug-ins.

**FRDocDidInsertPages**

The document was inserted some pages. This notification will be broadcast to all plug-ins.

**FRDocDidResizePage**

The page of [FR\\_Document](#) whose attribute was modified.

**FRDocDidRotatePage**

The page of [FR\\_Document](#) whose rotation attribute was modified.

**FRDocDoPassWordEncrypt**

Set the document password through the password input dialog.

**FRDocDoPrint**

Performs the print operation, including user dialog box.

**FRDocDoPrintSilently**

Performs the print operation, not including user dialog box.

**FRDocDoSave**

Saves a file, handling any user interface(for example, a Save File dialog box) if need.

**FRDocDoSave2**

Saves a file, handling any user interface(for example, a Save File dialog box) if need.

**FRDocDoSaveAs**

Displays a file dialog box which can be used to save the document as a new name.

**FRDocDoSaveAs2**

Displays a file dialog box which can be used to save the document as a new name.

**FRDocEnableRunScript**

Whether the document can run script.

**FRDocForbidChangeMark**

Forbid setting the modify flag. Reader has a built-in flag that indicate whether a document has been modified, if the value of the flag is valid, the Save button is enable, otherwise the Save button is disable.

**FRDocFromPDDoc**

Gets the [FR\\_Docuemnt](#) associated with a [FPD\\_Document](#).

**FRDocGenerateRedactions**

Generates a redacted document and it will be saved to *wsFilePath*.

**FRDocGenerateUR3Permission**

Generates the UR3 signature. Usage rights signatures are used to enable additional interactive features that are not available by default in a particular viewer application (such as Adobe Reader). See PDF reference for more details.

**FRDocGetAppPermissions**

Checks whether the user has the specified application permission or not.

**FRDocGetAppPermissionsII**

Gets the application permissions.

**FRDocGetChangeMark**

Checks whether the document is modified.

**FRDocGetCreateDocSource**

Gets the source type of the document.

**FRDocGetCreateDocSourceFileName**

Gets the source file name.

**FRDocGetCreateDocSourceFilePath**

Gets the source file path.

**FRDocGetCurCapture**

Gets the current capture handler.

**FRDocGetCurCaptureType**

Gets the type of the current capture handler.

**FRDocGetCurrentDocView**

Gets the current showing [FR\\_DocView](#) for specified document.

**FRDocGetCurrentSecurityMethodName**

Gets the name of current security method.

**FRDocGetCurrentWndProvider**

Gets the current window provider.

**FRDocGetCurSelection**

Gets the current selection handler.

**FRDocGetCurSelectionType**

Gets the type of the current selection handler.

**FRDocGetDocFrameHandler**

Gets the frame handler associated with the document.

**FRDocGetDocShowTitle**

Gets the title shown as the main frame title and the document tab title.

**FRDocGetDocumentType**

Get the type of the document.

**FRDocGetDocView**

Gets the specified [FR\\_DocView](#) for specified document.

**FRDocGetFilePath**

Gets the file path of a document opened by Foxit Reader.

**FRDocGetMergedPermissions**

Checks whether the document has the permission.

**FRDocGetOriginalType**

Gets the original type of the document. The real format of the opened document is PDF, but its wrapper format may be PPDF.

**FRDocGetParser**

Gets the PDF file parser which is parsing current document.

**FRDocGetPDDoc**

Gets the [FPD\\_Document](#) to associated with the specified [FR\\_Document](#).

**FRDocGetPDFCreator**

Gets the PDF creator associated with current document.

**FRDocGetPermissions**

Gets permissions of a document.

**FRDocGetPermissionsII**

Checks whether the document has the permission.

**FRDocGetReviewType**

Gets the review type.

**FRDocGetSignaturePermissions**

Gets the signature permissions of a document.

**FRDocGetTextSelectTool**

Gets the current text select tool for specified document.

**FRDocGetUIParentWnd**

Gets the UI parent window.

**FRDocGetWndProviderByName**

Gets the window provider by name.

**FRDocHasRedactAnnot**

Checks whether the document is marked for redaction or not.

**FRDocIsEnableRunScript**

Whether the document can run script.

**FRDocIsEncrypted**

When this interface is invoked, the [FRSecurityMethodIsMyMethod](#) will be invoked.

**FRDocIsImageBasedDocument**

Check if the document is Image Based Document.

**FRDocIsInProtectedViewMode**

Check if the document is in protected view mode.

**FRDocIsMemoryDoc**

Whether the document is a memory document or not.

**FRDocIsShowInBrowser**

Checks whether the document is opened in browser.

**FRDocIsValidAnnot**

Checks whether the annotation is valid or not.

**FRDocIsVisible**

Checks whether the opened document is visible or not.

**FRDocIsWillReopen**

Checks whether the document is to be reopened after it is closed.

**FRDocKillFocusAnnot**

Kills the focus annot.

**FRDocLoadAnnots**

Loads annotation(s) for an opening PDF document.

**FRDocOpenFromFile**

Opens and displays a document form a file.

**FRDocOpenFromFile2**

Opens and displays a document form a file.

**FRDocOpenFromPDDoc**

Opens and returns a [FR\\_Document](#) view of [PD\\_Document](#).

**FRDocOpenFromPDDoc2**

Opens and returns a [FR\\_Document](#) view of [PD\\_Document](#).

**FRDocParsePage**

Parses the specified page.

**FRDocPrintPages**

Sets the first page and the last page to be printed.

**FRDocPrintSetup**

Sets up the print.

**FRDocReleaseCurCapture**

Releases the current capture.

**FRDocReloadPage**

Reloads the specified page.

**FRDocRemoveFromSelection**

Removes some data from the current selection data and returns the updated one.

**FRDocRemoveSecurityMethod**

When this interface is invoked, the [FRSecurityMethodRemoveSecurityInfo](#) will be invoked.

**FRDocResetDocTitleColor**

Resets the document title color in the document tab.

**FRDocSetAppPermissions**

Sets the application permission.

**FRDocSetChangeMark**

Sets the modify flag. Reader has a built-in flag that indicate whether a document has been modified, if the value of the flag is valid, the Save button on File toolbar is enable, otherwise the Save button is disable.

**FRDocSetCreateDocSource**

Sets the source type of the document.

**FRDocSetCurCapture**

Set the current capture handler by type.

**FRDocSetCurrentWndProvider**

Sets the window provider.

**FRDocSetCurSelection**

Sets the current selection handler by type.

**FRDocSetCustomSecurity**

Sets security using custom security handler and custom encryption. Application should provide a full encryption dictionary (application can destroy it after this call), and a custom encryption handler.

**FRDocSetDocEncrypted**

Indicates whether the document is encrypted or not.

**FRDocSetDocTitleColor**

Sets the document title color in the document tab.

**FRDocSetDRMSecurity**

Sets security using custom security handler and custom encryption. Application should provide a full encryption dictionary (application can destroy it after this call), and a custom encryption handler.

**FRDocSetFocusAnnot**

Sets the focus annotation.

**FRDocSetInputPasswordProc**

Sets the prototype of callback function invoked by *Foxit Reader* to receive the password.

**FRDocSetMenuEnableByName**

Set the menu enable or not.

**FRDocSetOriginalType**

Sets the original type of the document. The real format of the opened document is PDF, but its wrapper format may be PPDF.

**FRDocSetPermissions**

Sets permissions to a document.

**FRDocSetPropertiesFilePath**

Sets the file path that will be shown in the properties dialog.

**FRDocSetPropertiesPDFVersion**

Sets the PDF version that will be shown in the properties dialog.

**FRDocSetReviewType**

Sets the review type.

**FRDocShowSaveProgressCancelButton**

Sets to show the save progress cancel button or not.

**FRDocShowSelection**

Displays the current selection by calling the selection handler's [FRSelectionShowSelection](#) () callback. It does nothing if the document has no selection, or the current selection's handler has no [FRSelectionShowSelection](#) () callback. It only raises those exceptions raised by the selection handler's [FRSelectionShowSelection](#) () callback.

**FRDocUpdateSecurityMethod**

When this interface is invoked, the [FRSecurityMethodRemoveSecurityInfo](#) will be invoked if the document is encrypted.

**FRDocWillDeletePages**

The document will delete any pages. This notification will be broadcast to all plug-ins.

**FRDocWillInsertPages**

The document will be inserted some pages. This notification will be broadcast to all plug-ins.

**FRDocWillResizePage**

The pages of [FR\\_Document](#) whose attribute will be modified.

**FRDocWillRotatePage**

The pages of [FR\\_Document](#) whose rotation attribute will be modified.

## Functions detail

### FRDocAddToCurrentSelection

#### Syntax

```
void* FRDocAddToCurrentSelection (
    FR_Document doc,
    void* pCurData,
    void* pAddData
);
```

#### Description

Adds the new data to the current selection data and returns the updated one.

#### Parameter

---

doc	[In] The input document.
-----	--------------------------

---

pCurData	[In] The current selection data.
----------	----------------------------------

---

pAddData	[In] The data to be added to the current selection handler.
----------	---

---

**Return**

The updated selection data. It is [NULL](#) if failed.

**Head file reference**

fr\_docTempl.h: 381

**FRDocCanSecurityMethodBeModified****Syntax**

```
FS_BOOL FRDocCanSecurityMethodBeModified (
    FR\_Document doc
);
```

**Description**

Whether the security applied to the document can be modified or not.

**Parameter**


---

doc	[In] The document.
-----	--------------------

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_docTempl.h: 572

**Related method**

[FRAppRegisterSecurityMethod](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRDocChangeDocShowTitle****Syntax**

```
void FRDocChangeDocShowTitle (
    FR\_Document doc,
    FS\_LPCWSTR lpwsShowTitle
);
```

**Description**

Sets the main frame title and the document tab title.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

lpwsShowTitle	[In] This value will be shown as the main frame title and the document tab title.
---------------	---

---

**Return**

void.

**Head file reference**

fr\_docTempl.h: 637

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRDocCheckInDocumentOLE****Syntax**

```
FS_BOOL FRDocCheckInDocumentOLE (
    FR Document doc
);
```

**Description**

Checks whether the document is opened by OLE. For example, the document is embedded in *MS Office Word*.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

True means the document is opened by OLE.

**Head file reference**

fr\_docTempl.h: 849

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRDocClearChangeMark****Syntax**

```
void FRDocClearChangeMark (
    FR\_Document doc
);
```

**Description**

Invalidates all modification.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 139

**Related method**

[FRDocGetChangeMark](#)

[FRDocSetChangeMark](#)

**FRDocClearSelection****Syntax**

```
void FRDocClearSelection (
    FR\_Document doc
);
```

**Description**

Clears and destroys the current selection by calling the appropriate selection server's [FRSelectionLosingSelection](#) ().

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 433

**FRDocClose****Syntax**

```
FS_BOOL FRDocClose (
    FR\_Document doc,
```

---

```
FS_BOOL bPromptToSave,
FS_BOOL bDelay,
FS_BOOL bShowCancel
);
```

**Description**

Closes the document window.

**Parameter**

doc	[In] The document to close.
bPromptToSave	[In] If <a href="#">FALSE</a> , the document closed without prompting the user and without saving, even if the document has been modified. If <a href="#">TRUE</a> , it prompt the user to save the document if it has been modified.
bDelay	[In] Delay closing the document or not.
bShowCancel	[In] If <a href="#">FALSE</a> , it will not show cancel button when prompt to save.

**Return**

[TRUE](#) if the document closed, [FALSE](#) if it did not. The document will always close if *bPromptToSave* is [FALSE](#) .

**Head file reference**

fr\_docTempl.h: 29

**Related method**

[FRDocOpenFromFile](#)

[FRDocOpenFromPDDoc](#)

**FRDocConvertPdfToOtherFormat****Syntax**

```
FS_BOOL FRDocConvertPdfToOtherFormat (
    FS_LPCWSTR wsSrcPath,
    FS_LPCWSTR wsDesPath,
    FS_DWordArray pageAry,
    FS_LPCWSTR wsFileExt,
    FS_LPCWSTR wsPwd,
    BOOL bShowProgress
);
```

**Description**

Convert Pdf document to other format document.

**Parameter**


---

wsSrcPath	[In] The source file which need to be converted.
wsDesPath	[In] The destination file path.
pageAry	[In] The pages need to be converted. if it is NULL ,will convert the whole document.
wsFileExt	[In] The format need to be convert. Support: docx, doc, xlsx, xls, html, pptx, rtf.
wsPwd	[In] The password of encrypt the document. if it is empty, will not encrypt the document.
bShowProgress	[In] Whether or not show the convert progress bar.

---

**Return**

[TRUE](#) converted sucessful.

**Head file reference**

fr\_docTempl.h: 1169

**Since**

[SDK LATEEST VERSION > 8.3.1](#)

**FRDocCopySelection****Syntax**

```
void FRDocCopySelection (
    FR\_Document doc
);
```

**Description**

Copies the current selection to the clipboard, if possible. The selection is copied if the selection handler has a [FRSelectionDoCopy](#) () callback, and the selection handler's [FRSelectionCanCopy](#) () callback returns [TRUE](#) . If the selection server does not have a [FRSelectionCanCopy](#) () method, a default value of [TRUE](#) is used. The selection is copied by calling the selection handler's [FRSelectionDoCopy](#) () callback. It only raises those exceptions raised by the selection handler's [FRSelectionDoCopy](#) () and [FRSelectionCanCopy](#) () callbacks.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 458

**FRDocCountDocViews****Syntax**

```
FS_INT32 FRDocCountDocViews (
    FR\_Document doc
);
```

**Description**

Gets the number of [FR\\_DocView](#) for specified document.

**Parameter**

---

doc	[In] The document whose view count is obtained.
-----	---

---

**Return**

The number of [FR\\_DocView](#) for specified document.

**Head file reference**

fr\_docTempl.h: 307

**Related method**

[FRDocGetDocView](#)

**FRDocCreateNewViewByWndProvider****Syntax**

```
void FRDocCreateNewViewByWndProvider (
    FR\_Document doc,
    FS\_LPCSTR lpsName
);
```

**Description**

Creates the new view by [FR\\_WndProviderCallbacksRec](#).

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

---

IpsName	[In] The Specified name of the window provider.
---------	---

---

**Return**

void.

**Head file reference**

fr\_docTempl.h: 889

**Related method**[FRAppRegisterWndProvider](#)**Since**[SDK LATEEST VERSION > 2.1.0.4](#)**FRDocDeleteSelection****Syntax**

```
void FRDocDeleteSelection (
    FR\_Document doc
);
```

**Description**

Deletes the specified document's current selection, if possible. The selection is deleted if changing the selection is currently permitted, the selection handler has an [FRSelectionDoDelete](#) () callback, and the selection server's [FRSelectionCanDelete](#) () callback returns [TRUE](#) . If the selection handler does not have a [FRSelectionCanDelete](#) () callback, a default value of [TRUE](#) is used. The selection is deleted by calling the selection handler's [FRSelectionDoDelete](#) () callback. It only raises those exceptions raised by the selection handler's [FRSelectionDoDelete](#) () and [FRSelectionCanDelete](#) () callbacks.

**Parameter**


---

doc	[In] The input document.
-----	--------------------------

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 443

**FRDocDidDeletePages****Syntax**

```
void FRDocDidDeletePages (
    FR\_Document doc,
    FS\_WordArray arrDelPage
);
```

**Description**

The document was deleted any pages. This notification will be broadcast to all plug-ins.

**Parameter**


---

doc	[In] The document whose page was deleted.
arrDelPage	[In] The index of the pages that has been deleted.

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 208

**Note:** You can call this method after you deleted some page from a FPD\_Document object.

The callback DidDeletePages() described in struct FR\_DocEventCallbacksRec will be called by Reader. Notes:

**FRDocDidInsertPages****Syntax**

```
void FRDocDidInsertPages (
    FR\_Document doc,
    FS\_INT32 iInserAt,
    FS\_INT32 nCount
);
```

**Description**

The document was inserted some pages. This notification will be broadcast to all plug-ins.

**Parameter**


---

doc	[In] The document to be inserted pages into.
iInserAt	[In] The page index for first inserted page.
nCount	[In] The page count for all inserted page.

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 180

**Note:** You can call this method after you inserted some page into a FPD\_Document object.  
The callback DidInsertPages() described in struct FR\_DocEventCallbacksRec will be called by Reader.

### FRDocDidResizePage

#### Syntax

```
void FRDocDidResizePage (
    FR\_Document doc,
    FS\_INT32 iPage
);
```

#### Description

The page of [FR\\_Document](#) whose attribute was modified.

#### Parameter

---

doc	[In] The document whose page's attribute was modified.
-----	--

---

iPage	[In] The page index.
-------	----------------------

---

#### Return

void

#### Head file reference

fr\_docTempl.h: 258

**Note:** You can call this method after you changed some page's rotation attribute.  
The callback DidModifyPageAttribute() described in struct FR\_DocEventCallbacksRec will be called by Reader.

### FRDocDidRotatePage

#### Syntax

```
void FRDocDidRotatePage (
    FR\_Document doc,
    FS\_INT32 iPage
);
```

#### Description

The page of [FR\\_Document](#) whose rotation attribute was modified.

#### Parameter

---

doc	[In] The document whose page's rotation attribute was modified.
-----	---

---

iPage	[In] The page index.
-------	----------------------

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 234

**Note:** You can call this method after you changed some page's rotation attribute. The callback DidModifyPageAttribute() described in struct FR\_DocEventCallbacksRec will be called by Reader.

**FRDocDoPassWordEncrypt****Syntax**

```
void FRDocDoPassWordEncrypt (
    FR\_Document frDoc,
    FRPasswordEncryptProc proc
);
```

**Description**

Set the document password through the password input dialog.

**Parameter**

---

frDoc	[In] The document to protected.
-------	---------------------------------

---

proc	[In] Prototype of callback function invoked by <i>Foxit Reader</i> to receive the protecting status.
------	--

---

**Return**

void.

**Head file reference**

fr\_docTempl.h: 1184

**Since**

[SDK LATEEST VERSION > 9.1](#)

**FRDocDoPrint****Syntax**

```
void FRDocDoPrint (
```

```
FR\_Document doc  
);
```

**Description**

Performs the print operation, including user dialog box.

**Parameter**

---

doc	[In] The document to print.
-----	-----------------------------

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 270

**Related method**

[FRDocPrintPages](#)

[FRDocDoPrintSilently](#)

[FRDocPrintSetup](#)

**FRDocDoPrintSilently****Syntax**

```
void FRDocDoPrintSilently (  
    FR\_Document doc  
,
```

**Description**

Performs the print operation, not including user dialog box.

**Parameter**

---

doc	[In] The document to print.
-----	-----------------------------

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 276

**Related method**

[FRDocPrintPages](#)

[FRDocPrintSetup](#)

**FRDocDoSave**

**Syntax**

```
void FRDocDoSave (
    FR\_Document doc,
    FR\_DocSaveProc proc,
    void* pProcData,
    FS\_BOOL bShowProgressBar
);
```

**Description**

Saves a file, handling any user interface(for example, a Save File dialog box) if need.

**Parameter**


---

doc	[In] The document to be saved.
-----	--------------------------------

---

proc	[In] Callback function.
------	-------------------------

---

pProcData	[In] The client data. It will be passed to the save callback function.
-----------	--

---

bShowProgressBar	[In] Whether to show the progress bar or not.
------------------	---

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 107

**Related method**

[FRDocDoSaveAs](#)

[FRDocSetChangeMark](#)

**FRDocDoSave2****Syntax**

```
FS_BOOL FRDocDoSave2 (
    FR\_Document doc,
    FR\_DocSaveProc proc,
    void* pProcData,
    FS\_BOOL bShowProgressBar,
    FS\_BOOL bDoPDFOptimize
);
```

**Description**

Saves a file, handling any user interface(for example, a Save File dialog box) if need.

**Parameter**

doc	[In] The document to be saved.
proc	[In] Callback function.
pProcData	[In] The client data. It will be passed to the save callback function.
bShowProgressBar	[In] Whether to show the progress bar or not.
bDoPDFOptimize	[In] Whether to optimize the PDF or not.

**Return**

True for success, otherwise failure.

**Head file reference**

fr\_docTempl.h: 923

**Related method**

[FRDocDoSave](#)

[FRDocDoSaveAs](#)

[FRDocSetChangeMark](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 3.0.0.0](#)

**FRDocDoSaveAs****Syntax**

```
void FRDocDoSaveAs (
    FR\_Document doc
);
```

**Description**

Displays a file dialog box which can be used to save the document as a new name.

**Parameter**

doc	[In] The document.
-----	--------------------

**Return****Head file reference**

fr\_docTempl.h: 115

**Related method**

[FRDocDoSave](#)**FRDocDoSaveAs2****Syntax**

```
FS_BOOL FRDocDoSaveAs2 (
    FR_Document doc,
    FS_LPCWSTR pwszFilePath,
    FR_DocSaveAsProc proc,
    void* pProcData,
    FS_BOOL bSaveAsTempFile,
    FS_BOOL bShowProgressBar
);
```

**Description**

Displays a file dialog box which can be used to save the document as a new name.

**Parameter**

doc	[In] The document.
pwszFilePath	[In] The path where the document to be saved as.
proc	[In] Callback function.
pProcData	[In] The client data. It will be passed to the save-as callback function.
bSaveAsTempFile	[In] Sets it FALSE as default.
bShowProgressBar	[In] Whether to show the progress bar or not.

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_docTempl.h: 811

**Related method**

[FR\\_DocSaveAsProc](#)

**Since**

[SDK LATEEST VERSION > 2.1](#)

**FRDocEnableRunScript**

**Syntax**

```
void FRDocEnableRunScript (
    FR\_Document doc,
    FS\_BOOL bIsEnable
);
```

**Description**

Whether the document can run script.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

bIsEnable	[In] Whether the document can run script.
-----------	---

---

**Return**

void.

**Head file reference**

[fr\\_docTempl.h: 616](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRDocForbidChangeMark

**Syntax**

```
void FRDocForbidChangeMark (
    FR\_Document doc,
    FS\_BOOL bForbid
);
```

**Description**

Forbid setting the modify flag. Reader has a built-in flag that indicate whether a document has been modified, if the value of the flag is valid, the Save button is enable, otherwise the Save button is disable.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

bForbid	[In] Whether to forbid setting the modify flag or not.
---------	--

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 1042

#### Related method

[FRDocSetChangeMark](#)  
[FRDocGetChangeMark](#)  
[FRDocClearChangeMark](#)

#### Since

[SDK LATEEST VERSION > 7.3.0.0](#)

**FRDocFromPDDoc**

#### Syntax

```
FR_Document FRDocFromPDDoc (
    FPD_Document pddoc
);
```

#### Description

Gets the [FR\\_Docuemnt](#) associated with a [FPD\\_Document](#).

#### Parameter

pddoc	[In] The <a href="#">FR_Docuemnt</a> whose <a href="#">FR_Document</a> is to be returned.
-------	---

#### Return

The [FR\\_Docuemnt](#) if a [FR\\_Docuemnt](#) is already opened for this [FPD\\_Document](#), otherwise [NULL](#).

#### Head file reference

fr\_docTempl.h: 31

#### Related method

[FRDocOpenFromFile](#)  
[FRDocOpenFromPDDoc](#)  
[FRDocGetPDDoc](#)

**FRDocGenerateRedactions**

#### Syntax

```
FS_BOOL FRDocGenerateRedactions (
    FR_Document doc,
    FS_WideString* wsFilePath
);
```

#### Description

Generates a redacted document and it will be saved to *wsFilePath*.

**Parameter**


---

doc	[In] The input document.
-----	--------------------------

---

wsFilePath	[Out] It receives the path of the document redacted.
------------	--

---

**Return**

TRUE for success, otherwise for failure.

**Head file reference**

fr\_docTempl.h: 1019

**Since**

[SDK LATEEST VERSION > 7.1.0.0](#)

**FRDocGenerateUR3Permission****Syntax**

```
FS_BOOL FRDocGenerateUR3Permission (
    FR\_Document doc
);
```

**Description**

Generates the UR3 signature. Usage rights signatures are used to enable additional interactive features that are not available by default in a particular viewer application (such as Adobe Reader). See PDF reference for more details.

**Parameter**


---

doc	[In] The input document.
-----	--------------------------

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_docTempl.h: 998

**Since**

[SDK LATEEST VERSION > 7.1.0.0](#)

**FRDocGetAppPermissions****Syntax**

```
FS_BOOL FRDocGetAppPermissions (
    FR\_Document doc,
    FS\_INT32 iPermission
);
```

**Description**

Checks whether the user has the specified application permission or not.

**Parameter**

---

doc	[In] The document whose user permission is obtained.
-----	--

---

iPermission	[In] The specified permission.
-------------	--------------------------------

---

**Return**

[TRUE](#) if the user has the specified application permission.

**Head file reference**

fr\_docTempl.h: 736

**Since**

[SDK LATEEST VERSION > 1.0](#)

## FRDocGetAppPermissionsII

**Syntax**

```
FS_DWORD FRDocGetAppPermissionsII (
    FR\_Document doc
);
```

**Description**

Gets the application permissions.

**Parameter**

---

doc	[In] The document whose user permission is obtained.
-----	--

---

**Return**

The application permissions.

**Head file reference**

fr\_docTempl.h: 747

**Since**

[SDK LATEEST VERSION > 1.0](#)

## FRDocGetChangeMark

**Syntax**

```
FS_BOOL FRDocGetChangeMark (
    FR\_Document doc
);
```

**Description**

Checks whether the document is modified.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

**Return**

[TRUE](#) if the document has been modified, [FALSE](#) if no change.

**Head file reference**

fr\_docTempl.h: 138

**Related method**

[FRDocSetChangeMark](#)

[FRDocClearChangeMark](#)

**FRDocGetCreateDocSource****Syntax**

```
FRCREATEDOCSOURCE FRDocGetCreateDocSource (
    FR\_Document doc
);
```

**Description**

Gets the source type of the document.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

**Return**

The source type of the document.

**Head file reference**

fr\_docTempl.h: 1076

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1.0](#)

**FRDocGetCreateDocSourceFileName**

**Syntax**

```
void FRDocGetCreateDocSourceFileName (
    FR\_Document doc,
    FS\_WideString* outSourceFileName
);
```

**Description**

Gets the source file name.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

outSourceFileName	[Out] It receives the source file name.
-------------------	---

---

**Return**

void.

**Head file reference**

fr\_docTempl.h: 1098

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1.0](#)

**FRDocGetCreateDocSourceFilePath****Syntax**

```
void FRDocGetCreateDocSourceFilePath (
    FR\_Document doc,
    FS\_WideString* outSourceFilePath
);
```

**Description**

Gets the source file path.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

outSourceFilePath	[Out] It receives the source file path.
-------------------	---

---

**Return**

void.

**Head file reference**

fr\_docTempl.h: 1205

**Since**  
[SDK LATEEST VERSION > 9.1](#)

## FRDocGetCurCapture

### Syntax

```
void* FRDocGetCurCapture (
    FR\_Document doc
);
```

### Description

Gets the current capture handler.

### Parameter

---

doc	[In] The input document.
-----	--------------------------

---

### Return

The current capture handler.

### Head file reference

fr\_docTempl.h: 484

## FRDocGetCurCaptureType

### Syntax

```
FS_LPCSTR FRDocGetCurCaptureType (
    FR\_Document doc
);
```

### Description

Gets the type of the current capture handler.

### Parameter

---

doc	[In] The input document.
-----	--------------------------

---

### Return

The type of the current capture handler.

### Head file reference

fr\_docTempl.h: 493

## FRDocGetCurrentDocView

**Syntax**

```
FR_DocView FRDocGetCurrentDocView (
    FR\_Document doc
);
```

**Description**

Gets the current showing [FR\\_DocView](#) for specified document.

**Parameter**

---

doc	[In] The document whose document view is obtained.
-----	--

---

**Return**

The current showing [FR\\_DocView](#).

**Head file reference**

fr\_docTempl.h: 323

**Related method**

[FRDocGetDocView](#)

**FRDocGetCurrentSecurityMethodName****Syntax**

```
void FRDocGetCurrentSecurityMethodName (
    FR\_Document doc,
    FS\_WideString* outName
);
```

**Description**

Gets the name of current security method.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

---

outName	[Out] It receives the name of current security method.
---------	--

---

**Return**

The name of current security method.

**Head file reference**

fr\_docTempl.h: 658

**Since**

[SDK LATEEST VERSION > 1.0](#)

## FRDocGetCurrentWndProvider

### Syntax

```
void* FRDocGetCurrentWndProvider (
    FR\_Document doc
);
```

### Description

Gets the current window provider.

### Parameter

---

doc	[In] The document.
-----	--------------------

---

### Return

The pointer to the current window provider.

### Head file reference

fr\_docTempl.h: 669

### Related method

[FRAppRegisterWndProvider](#)

### Since

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRDocGetCurSelection

### Syntax

```
void* FRDocGetCurSelection (
    FR\_Document doc
);
```

### Description

Gets the current selection handler.

### Parameter

---

doc	[In] The input document.
-----	--------------------------

---

### Return

The current selection handler.

### Head file reference

fr\_docTempl.h: 403

## FRDocGetCurSelectionType

**Syntax**

```
FS_LPCSTR FRDocGetCurSelectionType (
    FR\_Document doc
);
```

**Description**

Gets the type of the current selection handler.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

The type of the current selection handler.

**Head file reference**

[fr\\_docTempl.h](#): 412

**FRDocGetDocFrameHandler****Syntax**

```
HWND FRDocGetDocFrameHandler (
    FR\_Document doc
);
```

**Description**

Gets the frame handler associated with the document.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

The frame handler associated with the document.

**Head file reference**

[fr\\_docTempl.h](#): 879

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRDocGetDocShowTitle****Syntax**

```
void FRDocGetDocShowTitle (
    FR\_Document doc,
    FS\_WideString* outTitle
```

);

**Description**

Gets the title shown as the main frame title and the document tab title.

**Parameter**

doc	[In] The document.
outTitle	[Out] It receives the title shown as the main frame title and the document tab title.

**Return**

void.

**Head file reference**

fr\_docTempl.h: 912

**Since**

[SDK\\_LATEEST\\_VERSION > 3.0.0.0](#)

**FRDocGetDocumentType****Syntax**

```
FS_INT32 FRDocGetDocumentType (
    FR\_Document doc
);
```

**Description**

Get the type of the document.

**Parameter**

doc	[In] The document to get the type.
-----	------------------------------------

**Return**

The type of the document.

**Head file reference**

fr\_docTempl.h: 561

**Related method**

[FRDocTypes](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRDocGetDocView

### Syntax

```
FR_DocView FRDocGetDocView (
    FR\_Document doc,
    FS\_INT32 iView
);
```

### Description

Gets the specified [FR\\_DocView](#) for specified document.

### Parameter

---

doc	[In] The document whose document view is obtained.
-----	--

---

iView	[In] The index of a document view. The index range is 0 to ( <a href="#">FRDocCountDocViews</a> ()-1).
-------	--

---

### Return

The specified [FR\\_DocView](#).

### Head file reference

fr\_docTempl.h: 312

### Related method

[FRDocGetCurrentDocView](#)  
[FRDocCountDocViews](#)

## FRDocGetFilePath

### Syntax

```
void FRDocGetFilePath (
    FR\_Document doc,
    FS\_WideString* path
);
```

### Description

Gets the file path of a document opened by Foxit Reader.

### Parameter

---

doc	[In] The document whose file path is set.
-----	---

---

path	[In/Out] A wide string object to receive the file path.
------	---

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 360

**FRDocGetMergedPermissions**

**Syntax**

```
FS_BOOL FRDocGetMergedPermissions (
    FR\_Document doc,
    FS\_INT32 iPermission
);
```

**Description**

Checks whether the document has the permission.

**Parameter**

---

doc	[In] The document whose user permission is obtained.
-----	--

---

iPermission	[In] The specified permission.
-------------	--------------------------------

---

**Return**

[TRUE](#) means the document has the permission, otherwise not.

**Head file reference**

fr\_docTempl.h: 768

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRDocGetOriginalType**

**Syntax**

```
FROriginalDocType FRDocGetOriginalType (
    FR\_Document doc
);
```

**Description**

Gets the original type of the document. The real format of the opened document is PDF, but its wrapper format may be PPDF.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

The original type of the document. The real format of the opened document is PDF, but its wrapper format may be PPDF.

**Head file reference**

fr\_docTempl.h: 963

**Related method**

[FRDocSetOriginalType](#)

**Since**

[SDK LATEST VERSION > 3.0.0.0](#)

**FRDocGetParser****Syntax**

```
FPD_Parser FRDocGetParser (
    FR\_Document doc
);
```

**Description**

Gets the PDF file parser which is parsing current document.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

A PDF file parser associated with current document.

**Head file reference**

fr\_docTempl.h: 522

**FRDocGetPDDoc****Syntax**

```
FPD_Document FRDocGetPDDoc (
    FR\_Document doc
);
```

**Description**

Gets the [FPD\\_Document](#) to associated with the specified [FR\\_Document](#).

**Parameter**

---

doc	[In] The document whose <a href="#">FPD_Document</a> is obtained.
-----	---

---

**Return**

The [FPD\\_Document](#) associated with [FR\\_Document](#) .

**Head file reference**

fr\_docTempl.h: 58

**Related method**

[FRDocFromPDDoc](#)

[FRDocOpenFromPDDoc](#)

[FRPageViewGetPDPPage](#)

**FRDocGetPDFCreator****Syntax**

```
FPD_Creator FRDocGetPDFCreator (
    FR\_Document doc
);
```

**Description**

Gets the PDF creator associated with current document.

**Parameter**


---

doc	[In] The input document.
-----	--------------------------

---

**Return**

A PDF creator associated with current document.

**Head file reference**

fr\_docTempl.h: 531

**FRDocGetPermissions****Syntax**

```
FS_DWORD FRDocGetPermissions (
    FR\_Document doc
);
```

**Description**

Gets permissions of a document.

**Parameter**


---

doc	[In] The document whose user permission is obtained.
-----	--

---

**Return**

The document permissions.

**Head file reference**

fr\_docTempl.h: 339

**Related method**

[FRDocSetPermissions](#)

**FRDocGetPermissionsII****Syntax**

```
FS_BOOL FRDocGetPermissionsII (
    FR\_Document doc,
    FS_INT32 iPermission
);
```

**Description**

Checks whether the document has the permission.

**Parameter**

---

doc	[In] The document whose user permission is obtained.
-----	--

---

iPermission	[In] The input permission to check.
-------------	-------------------------------------

---

**Return**

[TRUE](#) means the document has the permission, otherwise not.

**Head file reference**

fr\_docTempl.h: 757

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRDocGetReviewType****Syntax**

```
FS_INT32 FRDocGetReviewType (
    FR\_Document doc
);
```

**Description**

Gets the review type.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

**Return**

The review type.

**Head file reference**

fr\_docTempl.h: 704

**Since**

[SDK LATEST VERSION > 1.0](#)

**FRDocGetSignaturePermissions****Syntax**

```
FS_BOOL FRDocGetSignaturePermissions (
    FR\_Document doc,
    pdwPermissions
);
```

**Description**

Gets the signature permissions of a document.

**Parameter**

---

doc	[In] The document whose user permission is obtained.
-----	--

---

---

pdwPermissions	[Out] The document permissions.
----------------	---------------------------------

---

**Return**

[TRUE](#) if the document is signed.

**Head file reference**

fr\_docTempl.h: 1158

**Since**

[SDK LATEST VERSION > 8.2.1](#)

**FRDocGetTextSelectTool****Syntax**

```
FR_TextSelectTool FRDocGetTextSelectTool (
    FR\_Document doc
);
```

**Description**

Gets the current text select tool for specified document.

**Parameter**

---

doc	[In] The document object.
-----	---------------------------

---

**Return**

The [FR\\_TextSelectTool](#) object of the *doc* .

**Head file reference**

fr\_docTempl.h: 551

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRDocGetUIParentWnd****Syntax**

```
void* FRDocGetUIParentWnd (
    FR\_Document doc
);
```

**Description**

Gets the UI parent window.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

A pointer to the parent window. It represents the *MFC CWnd\** .

**Head file reference**

fr\_docTempl.h: 869

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRDocGetWndProviderByName****Syntax**

```
void* FRDocGetWndProviderByName (
    FR\_Document doc,
    FS\_LPCSTR lpsName
);
```

**Description**

Gets the window provider by name.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

IpsName	[In] The Specified name of the window provider.
---------	---

---

**Return**

The window provider.

**Head file reference**

fr\_docTempl.h: 692

**Related method**

[FRAppRegisterWndProvider](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRDocHasRedactAnnot****Syntax**

```
FS_BOOL FRDocHasRedactAnnot (
    FR\_Document doc
);
```

**Description**

Checks whether the document is marked for redaction or not.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

TRUE if the document is marked for redaction, otherwise not.

**Head file reference**

fr\_docTempl.h: 1009

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

**FRDocIsEnableRunScript****Syntax**

```
FS_BOOL FRDocIsEnableRunScript (
    FR\_Document doc
```

);

**Description**

Whether the document can run script.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

**Return**

Whether the document can run script.

**Head file reference**

fr\_docTempl.h: 627

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRDocIsEncrypted****Syntax**

```
FS_BOOL FRDocIsEncrypted (
    FR\_Document doc
);
```

**Description**

When this interface is invoked, the [FRSecurityMethodIsMyMethod](#) will be invoked.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

**Return**

[TRUE](#) for the document being encrypted, otherwise not.

**Head file reference**

fr\_docTempl.h: 594

**Related method**

[FRAppRegisterSecurityMethod](#)

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRDocIsImageBasedDocument**

**Syntax**

```
FS_BOOL FRDocIsImageBasedDocument (
    FR\_Document frDoc
);
```

**Description**

Check if the document is Image Based Document.

**Parameter**

---

frDoc	[In] The document.
-------	--------------------

---

**Return**

[TRUE](#) means the document is Image Based Document.

**Head file reference**

fr\_docTempl.h: 1216

**Since**

[SDK\\_LATEEST\\_VERSION > 9.2](#)

**FRDocIsInProtectedViewMode****Syntax**

```
FS_BOOL FRDocIsInProtectedViewMode (
    FR\_Document frDoc
);
```

**Description**

Check if the document is in protected view mode.

**Parameter**

---

frDoc	[In] The document to protected.
-------	---------------------------------

---

**Return**

[TRUE](#) means in the protected view mode.

**Head file reference**

fr\_docTempl.h: 1195

**Since**

[SDK\\_LATEEST\\_VERSION > 9.1](#)

**FRDocIsMemoryDoc****Syntax**

```
FS_BOOL FRDocIsMemoryDoc (
    FR\_Document doc
);
```

**Description**

Whether the document is a memory document or not.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

**Return**

Whether the document is a memory document or not.

**Head file reference**

fr\_docTempl.h: 648

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRDocIsShowInBrowser****Syntax**

```
FS_BOOL FRDocIsShowInBrowser (
    FR\_Document doc
);
```

**Description**

Checks whether the document is opened in browser.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

True means the document is opened in browser.

**Head file reference**

fr\_docTempl.h: 859

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRDocIsValidAnnot****Syntax**

```
FS_BOOL FRDocIsValidAnnot (
```

```
FR_Document doc,  
FR_Annot frAnnot  
);
```

**Description**

Checks whether the annotation is valid or not.

**Parameter**

---

doc	[In] The document.
frAnnot	[In] The input annotation to be checked.

---

**Return**

TRUE if the annotation is valid.

**Head file reference**

fr\_docTempl.h: 1120

**Since**

[SDK LATEST VERSION > 7.3.1.0](#)

**FRDocIsVisible****Syntax**

```
FS_BOOL FRDocIsVisible (  
    FR_Document frDoc  
);
```

**Description**

Checks whether the opened document is visible or not.

**Parameter**

---

frDoc	[In] The document.
-------	--------------------

---

**Return**

Whether the opened document is visible or not.

**Head file reference**

fr\_docTempl.h: 1237

**Since**

[SDK LATEST VERSION > 9.3](#)

**FRDocIsWillReopen**

**Syntax**

```
FS_BOOL FRDocIsWillReopen (
    FR Document doc
);
```

**Description**

Checks whether the document is to be reopened after it is closed.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

**Return**

TRUE if the document is to be reopened.

**Head file reference**

fr\_docTempl.h: 1131

**Since**

[SDK\\_LATEEST\\_VERSION > 8.0.2](#)

**FRDocKillFocusAnnot****Syntax**

```
FS_BOOL FRDocKillFocusAnnot (
    FR Document doc
);
```

**Description**

Kills the focus annot.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_docTempl.h: 779

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRDocLoadAnnots****Syntax**

---

```
FS_BOOL FRDocLoadAnnots (
    FR\_Document doc,
    FS\_PtrArray arrAnnotDict
);
```

**Description**

Loads annotation(s) for an opening PDF document.

**Parameter**


---

doc	[In] The input document.
arrAnnotDict	[In] The pointer array holding the annotation dictionaries to be loaded.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fr\_docTempl.h: 901

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRDocOpenFromFile****Syntax**

```
FR_Document FRDocOpenFromFile (
    FS\_LPCWSTR wszFile,
    FS\_LPCSTR szPassword,
    FS\_BOOL bMakeVisible,
    FS\_BOOL bAddToMRU
);
```

**Description**

Opens and displays a document from a file.

**Parameter**


---

wszFile	[In] The file path.
szPassword	[In] The password with the file. It could be <a href="#">NULL</a> if no password to pass.
bMakeVisible	[In] Whether the document will be made visible.

---

---

bAddToMRU	[In] Whether add the file path to the recent file list.
-----------	---

---

**Return**

The document that was opened. It returns [NULL](#) if Reder failed to open document.

**Head file reference**

fr\_docTempl.h: 21

**Related method**

[FRDocClose](#)

[FRDocOpenFromPDDoc](#)

[FRDocFromPDDoc](#)

**Note:** Do not open and then immediately close a FR\_Document without letting least once event loop complete.

**FRDocOpenFromFile2****Syntax**

```
FR_Document FRDocOpenFromFile2 (
    FS\_LPCWSTR wszFile,
    FS LPCSTR szPassword,
    FS\_BOOL bMakeVisible,
    FS\_BOOL bAddToMRU,
    BOOL bCheckCertPassword
);
```

**Description**

Opens and displays a document form a file.

**Parameter**


---

wszFile	[In] The file path.
---------	---------------------

---

szPassword	[In] The password with the file. It could be <a href="#">NULL</a> if no password to pass.
------------	---

---

bMakeVisible	[In] Whether the document will be made visible.
--------------	---

---

bAddToMRU	[In] Whether add the file path to the recent file list.
-----------	---

---

bCheckCertPassword	[In] Whether prompt a password input dialog to check the password of the certificate used to encrypt the document.
--------------------	--

---

**Return**

The document that was opened. It returns [NULL](#) if Reder failed to open document.

**Head file reference**

fr\_docTempl.h: 1141

**Related method**[FRDocClose](#)[FRDocOpenFromPDDoc](#)[FRDocFromPDDoc](#)

**Note:** Do not open and then immediately close a FR\_Document without letting least once event loop complete.

**Since**[SDK\\_LATEEST\\_VERSION > 8.1](#)**FRDocOpenFromPDDoc****Syntax**

```
FR_Document FRDocOpenFromPDDoc (
    FPD\_Document pddoc,
    FS\_LPCWSTR lpwsTitle
);
```

**Description**

Opens and returns a [FR\\_Document](#) view of [PD\\_Document](#).

**Parameter**

---

pddoc	[In] The <a href="#">FR_Document</a> object to open.
-------	--

---

lpwsTitle	[In] It is used as window's title.
-----------	------------------------------------

---

**Return**

The document that was opened. It returns [NULL](#) if Reder failed to open document.

**Head file reference**

fr\_docTempl.h: 30

**Related method**[FRDocClose](#)[FRDocOpenFromFile](#)[FRDocFromPDDoc](#)

**Note:** Do not open and then immediately close a FR\_Document without letting least once event loop complete. FRDocClose() should be used in place of FPDDocClose() after FRDocOpenFromPDDoc() is called. FRDocClose() will decrement the FPDDoc appropriately and free document-related resources.

**FRDocOpenFromPDDoc2****Syntax**

```
FR_Document FRDocOpenFromPDDoc2 (
    FPD Document pddoc,
    FS\_LPCWSTR lpwsTitle,
    FS\_BOOL bDelPDFDoc
);
```

**Description**

Opens and returns a [FR\\_Document](#) view of [PD\\_Document](#).

**Parameter**

pddoc	[In] The <a href="#">FR_Document</a> object to open.
lpwsTitle	[In] It is used as window's title.
bDelPDFDoc	[In] Whether the framework must delete the <a href="#">FPD Document</a> object or not. If the plug-in deletes the <a href="#">FPD Document</a> object through <a href="#">FPDParserDestroy</a> (), sets it FALSE.

**Return**

The document that was opened. It returns [NULL](#) if Reder failed to open document.

**Head file reference**

fr\_docTempl.h: 1058

**Related method**

[FRDocClose](#)  
[FRDocOpenFromFile](#)  
[FRDocFromPDDoc](#)

**Note:**Do not open and then immediately close a FR\_Document without letting least once event loop complete. FRDocClose() should be used in place of FPDDocClose() after FRDocOpenFromPDDoc() is called. FRDocClose() will decrement the FPDDoc appropriately and free document-related resources.

**Since**

[SDK LATEEST VERSION > 7.3.1.0](#)

**FRDocParsePage****Syntax**

```
FS_BOOL FRDocParsePage (
    FR\_Document doc,
```

```
FS_INT32 nPageIndex  
);
```

**Description**

Parses the specified page.

**Parameter**

doc	[In] The document.
nPageIndex	[In] The specified page index.

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_docTempl.h: 1109

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1.0](#)

**FRDocPrintPages****Syntax**

```
void FRDocPrintPages (  
    FR_Document doc,  
    FS_INT32 firstPage,  
    FS_INT32 lastPage  
) ;
```

**Description**

Sets the first page and the last page to be printed.

**Parameter**

doc	[In] The input document.
firstPage	[In] The index of first page to be printed.
lastPage	[In] The index of last page to be printed.

**Return**

void

**Head file reference**

fr\_docTempl.h: 275

**Related method**

[FRDocDoPrint](#)

[FRDocDoPrintSilently](#)

**FRDocPrintSetup****Syntax**

```
void FRDocPrintSetup (
    FR\_Document doc
);
```

**Description**

Sets up the print.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 277

**Related method**

[FRDocDoPrint](#)

[FRDocDoPrintSilently](#)

[FRDocPrintPages](#)

**FRDocReleaseCurCapture****Syntax**

```
void FRDocReleaseCurCapture (
    FR\_Document doc
);
```

**Description**

Releases the current capture.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 502

**FRDocReloadPage****Syntax**

```
FS_BOOL FRDocReloadPage (
    FR_Document doc,
    FS_INT32 iPage,
    FS_BOOL bDisableGoto
);
```

**Description**

Reloads the specified page.

**Parameter**

doc	[In] The input document.
iPage	[In] The specified page index.
bDisableGoto	[In] Whether to prevent going to the specified page view. Sets it FALSE as default.

**Return**

TRUE for success, otherwise for failure.

**Head file reference**

fr\_docTempl.h: 1030

**Since**

[SDK LATEEST VERSION > 7.2.2](#)

**FRDocRemoveFromSelection****Syntax**

```
void* FRDocRemoveFromSelection (
    FR_Document doc,
    void* pCurData,
    void* pRemData
);
```

**Description**

Removes some data from the current selection data and returns the updated one.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

pCurData	[In] The current selection data.
----------	----------------------------------

---

pRemData	[In] The data to be removed from the current selection data.
----------	--

---

**Return**

The updated selection data. It is [NULL](#) if failed.

**Head file reference**

fr\_docTempl.h: 392

**FRDocRemoveSecurityMethod****Syntax**

```
FS_BOOL FRDocRemoveSecurityMethod (
    FR\_Document doc
);
```

**Description**

When this interface is invoked, the [FRSecurityMethodRemoveSecurityInfo](#) will be invoked.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

**Return**

[TRUE](#) for the security method of the document being removed, otherwise not.

**Head file reference**

fr\_docTempl.h: 605

**Related method**

[FRAppRegisterSecurityMethod](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRDocResetDocTitleColor****Syntax**

```
void FRDocResetDocTitleColor (
```

```
FR_Document doc  
);
```

**Description**

Resets the document title color in the document tab.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 940

**Related method**

[FRDocSetDocTitleColor](#)

**Since**

[SDK LATEEST VERSION > 3.0.0.0](#)

**FRDocSetAppPermissions****Syntax**

```
void FRDocSetAppPermissions (  
    FR_Document doc,  
    FS_DWORD dwPermission  
)
```

**Description**

Sets the application permission.

**Parameter**

---

doc	[In] The document whose user permission is obtained.
-----	--

---

dwPermission	[In] The input application permission.
--------------	--

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 725

**Since**

[SDK LATEEST VERSION > 1.0](#)

## FRDocSetChangeMark

### Syntax

```
void FRDocSetChangeMark (
    FR\_Document doc
);
```

### Description

Sets the modify flag. Reader has a built-in flag that indicate whether a document has been modified, if the value of the flag is valid, the Save button on File toolbar is enable, otherwise the Save button is disable.

### Parameter

---

doc	[In] The document.
-----	--------------------

---

### Return

void

### Head file reference

fr\_docTempl.h: 116

### Related method

[FRDocGetChangeMark](#)

[FRDocClearChangeMark](#)

## FRDocSetCreateDocSource

### Syntax

```
void FRDocSetCreateDocSource (
    FR\_Document doc,
    FRCREATEDOCSOURCE sourceType,
    FS\_LPCWSTR lpwsSourceFileName
);
```

### Description

Sets the source type of the document.

### Parameter

---

doc	[In] The document.
-----	--------------------

---

sourceType	[In] The input source type of the document.
------------	---

---

lpwsSourceFileName	[In] The input source file name.
--------------------	----------------------------------

---

**Return**

void.

**Head file reference**

fr\_docTempl.h: 1086

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1.0](#)

**FRDocSetCurCapture****Syntax**

```
FS_BOOL FRDocSetCurCapture (
    FR\_Document doc,
    FS\_LPSTR type,
    void* pCapData
);
```

**Description**

Set the current capture handler by type.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

type	[In] The type of the capture handler.
------	---------------------------------------

---

pCapData	[In] The capture data.
----------	------------------------

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_docTempl.h: 473

**FRDocSetCurrentWndProvider****Syntax**

```
void FRDocSetCurrentWndProvider (
    FR\_Document doc,
    void* pProvider
);
```

**Description**

Sets the window provider.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

pProvider	[In] The input window provider.
-----------	---------------------------------

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 680

**Related method**[FRAppRegisterWndProvider](#)**Since**[SDK LATEEST VERSION > 1.0](#)**FRDocSetCurSelection****Syntax**

```
FS_BOOL FRDocSetCurSelection (
    FR_Document doc,
    FS_LPSTR type,
    void* pSelectData
);
```

**Description**

Sets the current selection handler by type.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

type	[In] The input type of the selection handler.
------	---

---

pSelectData	[In] The selection data.
-------------	--------------------------

---

**Return**[TRUE](#) means successful, otherwise not.**Head file reference**

fr\_docTempl.h: 370

**FRDocSetCustomSecurity****Syntax**

```
void FRDocSetCustomSecurity (
    FR\_Document doc,
    FPD\_Object encryptDict,
    FR\_CryptoCallbacks cryptoHandler,
    FS\_BOOL bEncryptMetadata,
    FS\_LPVVOID clientHandler
);
```

**Description**

Sets security using custom security handler and custom encryption. Application should provide a full encryption dictionary (application can destroy it after this call), and a custom encryption handler.

**Parameter**

doc	[In] The input document.
encryptDict	[In] The Encrypt dictionary.
cryptoHandler	[In] The callbacks for crypto handler.
bEncryptMetadata	[In] Whether to encrypt the metadata.
clientHandler	[In] The user-supplied data.

**Return**

void

**Head file reference**

fr\_docTempl.h: 92

**FRDocSetDocEncrypted****Syntax**

```
void FRDocSetDocEncrypted (
    FR\_Document frDoc,
    FS\_BOOL bEncrypted
);
```

**Description**

Indicates whether the document is encrypted or not.

**Parameter**

---

frDoc	[In] The document.
-------	--------------------

---

bEncrypted	[In] Indicates whether the document is encrypted or not.
------------	--

---

**Return**

void.

**Head file reference**

fr\_docTempl.h: 1226

**Since**[SDK\\_LATEEST\\_VERSION > 9.2](#)**FRDocSetTitleColor****Syntax**

```
void FRDocSetTitleColor (
    FR\_Document doc,
    FS\_COLORREF clrDocTitle
);
```

**Description**

Sets the document title color in the document tab.

**Parameter**


---

doc	[In] The input document.
-----	--------------------------

---

clrDocTitle	[In] The input color value.
-------------	-----------------------------

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 946

**Related method**[FRDocResetDocTitleColor](#)**Since**[SDK\\_LATEEST\\_VERSION > 3.0.0.0](#)**FRDocSetDRMSecurity****Syntax**

```
void FRDocSetDRMSecurity (
```

---

```
FR\_Document doc,
FPD\_Object encryptDict,
FR\_DRMCryptoCallbacks cryptoHandler,
FS\_BOOL bEncryptMetadata,
FS\_LPVOID clientHandler
);
```

**Description**

Sets security using custom security handler and custom encryption. Application should provide a full encryption dictionary (application can destroy it after this call), and a custom encryption handler.

**Parameter**

doc	[In] The input document.
encryptDict	[In] The Encrypt dictionary.
cryptoHandler	[In] The callbacks for crypto handler.
bEncryptMetadata	[In] Whether to encrypt the metadata.
clientHandler	[In] The user-supplied data.

**Return**

void

**Head file reference**

fr\_docTempl.h: 1247

**Since**

[SDK LATEEST VERSION > 9.5](#)

**FRDocSetFocusAnnot****Syntax**

```
FS_BOOL FRDocSetFocusAnnot (
    FR\_Document doc,
    FR\_Annot pFocusAnnot,
    FS\_BOOL bDelay
);
```

**Description**

Sets the focus annotation.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

pFocusAnnot	[In] The input annotation.
-------------	----------------------------

---

bDelay	[In] Whether to delay the setting or not. The default value is FALSE.
--------	---

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 986

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

**FRDocSetInputPasswordProc****Syntax**

```
void FRDocSetInputPasswordProc (
    FR\_Document doc,
    FRInputPasswordProc proc
);
```

**Description**

Sets the prototype of callback function invoked by *Foxit Reader* to receive the password.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

proc	[In] Prototype of callback function invoked by <i>Foxit Reader</i> to receive the password.
------	---

---

**Return**

void.

**Head file reference**

fr\_docTempl.h: 838

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.2](#)

**FRDocSetMenuEnableByName**

**Syntax**

```
void FRDocSetMenuEnableByName (
    FR\_Document doc,
    FS\_LPCSTR csName,
    FS\_BOOL bEnable
);
```

**Description**

Set the menu enable or not.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

csName	[In] The menu you want to control.
--------	------------------------------------

---

bEnable	[In] Whether the menu is enable or not.
---------	---

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 511

**FRDocSetOriginalType****Syntax**

```
void FRDocSetOriginalType (
    FR\_Document doc,
    FROriginalDocType oriDocType
);
```

**Description**

Sets the original type of the document. The real format of the opened document is PDF, but its wrapper format may be PPDF.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

oriDocType	[In] The input original type of the document.
------------	---

---

**Return**

void.

**Head file reference**

fr\_docTempl.h: 969

**Related method**

[FRDocGetOriginalType](#)

**Since**

[SDK LATEST VERSION > 3.0.0.0](#)

## FRDocSetPermissions

**Syntax**

```
void FRDocSetPermissions (
    FR\_Document doc,
    FS\_DWORD dwPermission
);
```

**Description**

Sets permissions to a document.

**Parameter**

---

doc	[In] The document whose user permission is set.
-----	---

---

dwPermission	[In] The new permission to set to the document.
--------------	---

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 344

**Related method**

[FRDocGetPermissions](#)

## FRDocSetPropertiesFilePath

**Syntax**

```
void FRDocSetPropertiesFilePath (
    FR\_Document doc,
    FS\_LPCWSTR lpwsFilePath
);
```

**Description**

Sets the file path that will be shown in the properties dialog.

**Parameter**

doc	[In] The document.
lpwsFilePath	[In] The input file path that will be shown in the properties dialog.

**Return**

void

**Head file reference**

fr\_docTempl.h: 800

**Since**[SDK LATEEST VERSION > 2.0](#)**FRDocSetPropertiesPDFVersion****Syntax**

```
void FRDocSetPropertiesPDFVersion (
    FR\_Document doc,
    FS\_LPCWSTR lpwsPDFVersion
);
```

**Description**

Sets the PDF version that will be shown in the properties dialog.

**Parameter**

doc	[In] The document.
lpwsPDFVersion	[In] The input PDF version that will be shown in the properties dialog.

**Return**

void

**Head file reference**

fr\_docTempl.h: 789

**Since**[SDK LATEEST VERSION > 2.0](#)**FRDocSetReviewType****Syntax**

```
void FRDocSetReviewType (
    FR\_Document doc,
    FS\_INT32 nType
```



);

**Description**

Sets the review type.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

nType	[In] The input review type.
-------	-----------------------------

---

**Return**

void.

**Head file reference**

fr\_docTempl.h: 714

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRDocShowSaveProgressCancelButton****Syntax**

```
void FRDocShowSaveProgressCancelButton (
    FR\_Document doc,
    FS\_BOOL bShow
);
```

**Description**

Sets to show the save progress cancel button or not.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

bShow	[In] Whether to show the save progress cancel button.
-------	---

---

**Return**

void.

**Head file reference**

fr\_docTempl.h: 827

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1](#)

## FRDocShowSelection

### Syntax

```
void FRDocShowSelection (
    FR Document doc
);
```

### Description

Displays the current selection by calling the selection handler's [FRSelectionShowSelection](#) () callback. It does nothing if the document has no selection, or the current selection's handler has no [FRSelectionShowSelection](#) () callback. It only raises those exceptions raised by the selection handler's [FRSelectionShowSelection](#) () callback.

### Parameter

---

doc	[In] The input document.
-----	--------------------------

---

### Return

void

### Head file reference

fr\_docTempl.h: 421

## FRDocUpdateSecurityMethod

### Syntax

```
FS_BOOL FRDocUpdateSecurityMethod (
    FR Document doc
);
```

### Description

When this interface is invoked, the [FRSecurityMethodRemoveSecurityInfo](#) will be invoked if the document is encrypted.

### Parameter

---

doc	[In] The document.
-----	--------------------

---

### Return

[TRUE](#) for success, otherwise failure.

### Head file reference

fr\_docTempl.h: 583

### Related method

[FRAppRegisterSecurityMethod](#)

**Since**  
[SDK LATEEST VERSION > 1.0](#)

### FRDocWillDeletePages

#### Syntax

```
void FRDocWillDeletePages (
    FR\_Document doc,
    FS\_WordArray arrDelPage
);
```

#### Description

The document will delete any pages. This notification will be broadcast to all plug-ins.

#### Parameter

doc	[In] The document whose page was deleted.
-----	---

arrDelPage	[In] The index of the pages that has been deleted.
------------	--

#### Return

void

#### Head file reference

fr\_docTempl.h: 194

**Note:** You can call this method before you delete some page from a FPD\_Document object.

The callback WillDeletePages() described in struct FR\_DocEventCallbacksRec will be called by Reader. Notes:

### FRDocWillInsertPages

#### Syntax

```
void FRDocWillInsertPages (
    FR\_Document doc,
    FS\_INT32 iInserAt,
    FS\_INT32 nCount
);
```

#### Description

The document will be inserted some pages. This notification will be broadcast to all plug-ins.

#### Parameter

---

doc	[In] The document to be inserted pages into.
-----	--

---

iInserAt	[In] The page index for first inserted page.
----------	--

---

nCount	[In] The page count for all inserted page.
--------	--

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 166

**Note:** You can call this method before you start to insert some page into a FPD\_Document object.

The callback WillInsertPages() descripted in struct FR\_DocEventCallbacksRec will be called by Reader.

**FRDocWillResizePage****Syntax**

```
void FRDocWillResizePage (
    FR\_Document doc,
    int iPage
);
```

**Description**

The pages of [FR\\_Document](#) whose attribute will be modified.

**Parameter**


---

doc	[In] The document whose page's attribute was modified.
-----	--

---

iPage	[In] The page index.
-------	----------------------

---

**Return**

void

**Head file reference**

fr\_docTempl.h: 246

**Note:** You can call this method before you want to change some page's rotation attribute. The callback WillModifyPageAttribute() descripted in struct FR\_DocEventCallbacksRec will be called by Reader.

**FRDocWillRotatePage**

**Syntax**

```
void FRDocWillRotatePage (
    FR\_Document doc,
    int iPage
);
```

**Description**

The pages of [FR\\_Document](#) whose rotation attribute will be modified.

**Parameter**

doc	[In] The document whose page's rotation attribute was modified.
-----	---

iPage	[In] The page index.
-------	----------------------

**Return**

void

**Head file reference**

`fr_docTempl.h: 222`

**Note:** You can call this method before you want to change some page's rotation attribute. The callback WillModifyPageAttribute() described in struct `FR_DocEventCallbacksRec` will be called by Reader.

## FR\_DocView

### [Return from Used by](#)

**Description**

A [FR\\_DocView](#) is the area of the Foxit Reader window that displays the contents of a document page. Every [FR\\_DocView](#) may have more than one [FR\\_PageView](#). It contains references to the [FPD\\_Document](#) or the document being displayed.

[FR\\_DocView](#) has methods to display a page, select a zoom factor, scroll the page displayed inside, control screen redrawing, and traverse the history stack that records where users have been in a document.

To get a [FR\\_PageView](#) which displayed in a [FR\\_DocView](#), using [FRDocViewGetPageView](#) and [FRDocViewGetPageViewAtPoint](#).

**Returned from**

[FRDocGetCurrentDocView](#)  
[FRDocGetDocView](#)  
[FRPageViewGetDocView](#)

**Used by**

[FRTextSelectToolDrawSelection](#)  
[FRDocViewCountPageViews](#)  
[FRDocViewCountVisiblePageViews](#)  
[FRDocViewDoPopUpMenu](#)  
[FRDocViewDrawNow](#)  
[FRDocViewDrawRect](#)  
[FRDocViewGetCurrentPageView](#)  
[FRDocViewGetCurrentSnapshot](#)  
[FRDocViewGetDocument](#)  
[FRDocViewGetLayoutMode](#)  
[FRDocViewGetLRCDocViewText](#)  
[FRDocViewGetMachinePort](#)  
[FRDocViewGetMaxScrollingSize](#)  
[FRDocViewGetOCContext](#)  
[FRDocViewGetPageView](#)  
[FRDocViewGetPageViewAtPoint](#)  
[FRDocViewGetRotation](#)  
[FRDocViewGetTagDocViewText](#)  
[FRDocViewGetThumbnailView](#)  
[FRDocViewGetVisiblePageView](#)  
[FRDocViewGetZoom](#)  
[FRDocViewGetZoomType](#)  
[FRDocViewGoBack](#)  
[FRDocViewGoForward](#)  
[FRDocViewGotoPageView](#)  
[FRDocViewGotoPageViewByPoint](#)  
[FRDocViewGotoPageViewByRect](#)  
[FRDocViewGotoPageViewByState](#)  
[FRDocViewInvalidateRect](#)  
[FRDocViewIsValidPageView](#)  
[FRDocViewMovePage](#)  
[FRDocViewProcAction](#)  
[FRDocViewReleaseMachinePort](#)  
[FRDocViewScrollTo](#)  
[FRDocViewSetLayoutMode](#)  
[FRDocViewSetRotation](#)  
[FRDocViewShowAnnots](#)  
[FRDocViewZoomTo](#)

## Definitions

### Definitions summary

#### **FR\_ROTATE\_POS\_BOTTOM**

270 degree (clockwise).

#### **FR\_ROTATE\_POS\_LEFT**

180 degree (clockwise).

#### **FR\_ROTATE\_POS\_RIGHT**

90 degree (clockwise).

#### **FR\_ROTATE\_POS\_TOP**

0 degree (clockwise).

### Definitions detail

## FR\_ROTATE\_POS\_BOTTOM

### Syntax

```
#define FR_ROTATE_POS_BOTTOM 2
```

### Description

270 degree (clockwise).

### Group

[FR\\_RotationFlags](#)

### Head file reference

fr\_viewExpT.h: 128

## FR\_ROTATE\_POS\_LEFT

### Syntax

```
#define FR_ROTATE_POS_LEFT 3
```

### Description

180 degree (clockwise).

### Group

[FR\\_RotationFlags](#)

### Head file reference

fr\_viewExpT.h: 131

## FR\_ROTATE\_POS\_RIGHT

### Syntax

```
#define FR_ROTATE_POS_RIGHT 1
```

### Description

90 degree (clockwise).

### Group

[FR\\_RotationFlags](#)

### Head file reference

fr\_viewExpT.h: 125

## FR\_ROTATE\_POS\_TOP

### Syntax

```
#define FR_ROTATE_POS_TOP 0
```

**Description**

0 degree (clockwise).

**Group**

[FR\\_RotationFlags](#)

**Head file reference**

fr\_viewExpT.h: 122

## Enumerations

### Enumerations summary

**[FRDOCVIEW\\_LAYOUTMODE](#)**

A structure that defines the layout of a document. See [FRDocViewGetLayoutMode](#) .

**[FRDOCVIEW\\_ZOOMTYPE](#)**

Constants that specify the zoom strategy that Foxit Reader is to follow. See [FRDocViewGetZoomType](#) .

### Enumerations detail

**FRDOCVIEW\_LAYOUTMODE****Syntax**

```
enum FRDOCVIEW_LAYOUTMODE{
    FR\_LAYOUTMODE\_SINGLE,
    FR\_LAYOUTMODE\_CONTINUOUS,
    FR\_LAYOUTMODE\_FACING,
    FR\_LAYOUTMODE\_CONTINUOUS\_FACING
};
```

**Description**

A structure that defines the layout of a document. See [FRDocViewGetLayoutMode](#) .

**Head file reference**

fr\_viewExpT.h: 64

**FR\_LAYOUTMODE\_SINGLE**

Using single page mode.

**FR\_LAYOUTMODE\_CONTINUOUS**

Using one-column continue mode.

**FR\_LAYOUTMODE\_FACING**

Using facing mode.

**FR\_LAYOUTMODE\_CONTINUOUS\_FACING**

Using facing continue mode.

## FRDOCVIEW\_ZOOMTYPE

### Syntax

```
enum FRDOCVIEW_ZOOMTYPE{
    FR\_ZOOM\_MODE\_NONE,
    FR\_ZOOM\_MODE\_ACTUAL\_SCALE,
    FR\_ZOOM\_MODE\_ACTUAL\_SIZE,
    FR\_ZOOM\_MODE\_FIT\_PAGE,
    FR\_ZOOM\_MODE\_FIT\_WIDTH,
    FR\_ZOOM\_MODE\_FIT\_HEIGHT,
    FR\_ZOOM\_MODE\_FIT\_RECTANGLE,
    FR\_ZOOM\_MODE\_FIT\_BOUNDINGBOX,
    FR\_ZOOM\_MODE\_FIT\_VISIBLE,
    FR\_ZOOM\_MODE\_AUTOMATIC
};
```

### Description

Constants that specify the zoom strategy that Foxit Reader is to follow. See [FRDocViewGetZoomType](#).

#### Head file reference

fr\_viewExpT.h: 81

##### **FR\_ZOOM\_MODE\_NONE**

None zoom type.

##### **FR\_ZOOM\_MODE\_ACTUAL\_SCALE**

No variable zoom (the zoom is a fixed value such as 1.0 for 100%).

##### **FR\_ZOOM\_MODE\_ACTUAL\_SIZE**

Actual size.

##### **FR\_ZOOM\_MODE\_FIT\_PAGE**

To keep entire page visible.

##### **FR\_ZOOM\_MODE\_FIT\_WIDTH**

Fits the page width to the window.

##### **FR\_ZOOM\_MODE\_FIT\_HEIGHT**

Fits the page height to the window.

##### **FR\_ZOOM\_MODE\_FIT\_RECTANGLE**

Fits rectangle.

##### **FR\_ZOOM\_MODE\_FIT\_BOUNDINGBOX**

Fits bounding box.

##### **FR\_ZOOM\_MODE\_FIT\_VISIBLE**

Fits visible.

##### **FR\_ZOOM\_MODE\_AUTOMATIC**

Automatic.

## Structures

### Structures summary

#### [FRDESTINFO](#)

Doc dest info.

#### [FRLayoutInfo](#)

The display information for document view.

### Structs detail

#### FRDESTINFO

##### Syntax

```
typedef struct _fr_destinfo_{
    FS_INT32 m_nLeft,
    FS_INT32 m_nTop,
    FS_INT32 m_nWidth,
    FS_INT32 m_nHeight,
    FS_INT32 m_iPage,
    FS_INT32 m_nZoomToMode,
    double m_dbScale,
    FS_BOOL m_bPdfCord,
    FS_BOOL m_bLink
}FRDESTINFO, *PFRDESTINFO;
```

##### Description

Doc dest info.

##### Head file reference

fr\_viewExpT.h: 141

##### **m\_nLeft**

m\_nLeft

##### **m\_nTop**

m\_nTop

##### **m\_nWidth**

m\_nWidth

##### **m\_nHeight**

m\_nHeight

##### **m\_iPage**

m\_iPage

##### **m\_nZoomToMode**

m\_nZoomToMode

##### **m\_dbScale**

m\_dbScale

**m\_bPdfCord**  
m\_bPdfCord

**m\_bLink**  
m\_bLink

## FRLayoutInfo

### Syntax

```
typedef struct _page_layoutinfo_{
    FS_INT32 m_nRotate,
    FS_INT32 m_nZoomMode,
    FS_INT32 m_nShowMode,
    FS_INT32 m_nFacingCount,
    FS_INT32 m_nMarginSize,
    double m_dbZoom,
    FS_BOOL m_bDispGrid,
    FS_BOOL m_bReserved,
    FS_BOOL m_bReplaceColor,
    FS_BOOL m_bCoverPage,
    PFRDESTINFO m_pDestInfo
}FRLayoutInfo, *PFRLayoutInfo;
```

### Description

The display information for document view.

**Head file reference**  
fr\_viewExpT.h: 169

**m\_nRotate**  
m\_nRotate

**m\_nZoomMode**  
m\_nZoomMode

**m\_nShowMode**  
m\_nShowMode

**m\_nFacingCount**  
m\_nFacingCount

**m\_nMarginSize**  
m\_nMarginSize

**m\_dbZoom**  
m\_dbZoom

**m\_bDispGrid**  
m\_bDispGrid

**m\_bReserved**

m\_bReserved

**m\_bReplaceColor**  
m\_bReplaceColor

**m\_bCoverPage**  
m\_bCoverPage

**m\_pDestInfo**  
m\_pDestInfo

## Functions

### Functions summary

#### [\*\*FRDocViewCountPageViews\*\*](#)

Gets the number of page views for the specified document view.

#### [\*\*FRDocViewCountVisiblePageViews\*\*](#)

Gets the number of visible page views for specified document view.

#### [\*\*FRDocViewDoPopUpMenu\*\*](#)

Displays the given menu as a pop-up menu anchored at *xHit* and *yHit*, which are in device space coordinates relative to *docView*.

#### [\*\*FRDocViewDrawNow\*\*](#)

Forces specified document view to redraw.

#### [\*\*FRDocViewDrawRect\*\*](#)

Draws rectangle in the doc view.

#### [\*\*FRDocViewGetCurrentPageView\*\*](#)

Gets current page view that is visible in screen.

#### [\*\*FRDocViewGetCurrentSnapshot\*\*](#)

Gets current snapshot image that generated by using snapshot tool.

#### [\*\*FRDocViewGetDocument\*\*](#)

Gets the [FR\\_Document](#) associated with specified document view..

#### [\*\*FRDocViewGetLayoutMode\*\*](#)

Gets the page layout mode.

#### [\*\*FRDocViewGetLRCdocViewText\*\*](#)

Gets the text of a page in LR Order.

#### [\*\*FRDocViewGetMachinePort\*\*](#)

Gets the platform-specific object needed to draw into page view using a platform's native graphic calls. In Win32 OS, it is a [FR\\_WinPort](#) object which contain the *HWND* handler and the device context. When done, release it using [FRDocViewReleaseMachinePort](#) () .

#### [\*\*FRDocViewGetMaxScrollingSize\*\*](#)

Gets the maximum scrolling size of the document view.

#### [\*\*FRDocViewGetOCCContext\*\*](#)

Gets the *OCG* context of Reader doc view.

#### [\*\*FRDocViewGetPageView\*\*](#)

Gets the specified [FR\\_PageView](#) for specified document view.

#### [\*\*FRDocViewGetPageViewAtPoint\*\*](#)

Gets a specified [FR\\_PageView](#) with a point which is underlying to the area of page view.

#### [\*\*FRDocViewGetRotation\*\*](#)

Gets the rotation factor.

**FRDocViewGetTagDocViewText**

Gets the text of a page in Tag Order.

**FRDocViewGetThumbnailView**

Gets the thumbnail view related to the input document view.

**FRDocViewGetVisiblePageView**

Gets the specified visible page.

**FRDocViewGetZoom**

Gets the current zoom for page view. The zoom factor is point-to-point, not point-to-pixel.

**FRDocViewGetZoomType**

Gets the current zoom type.

**FRDocViewGoBack**

Goes to the previous view on the view stack, if a previous view exist.

**FRDocViewGoForward**

Goes to the next view on the view stack, if a next view exist.

**FRDocViewGotoPageView**

Goes to specified page, retaining the current display mode. It invalidates the display, but not always perform an immediately redraw.

**FRDocViewGotoPageViewByPoint**

Goes to specified page and scroll page view to the location specified by *left* and *top* , retaining the current display mode. It invalidates the display, but not always perform an immediately redraw.

**FRDocViewGotoPageViewByRect**

Goes to specified page view and scroll to center of the rectangle. It always change the zoom level if necessary.

**FRDocViewGotoPageViewByState**

Goes to the page view by state.

**FRDocViewInvalidateRect**

Redraws the specified area of document view immediately.

**FRDocViewIsValidPageView**

Checks whether the page view is valid or not.

**FRDocViewMovePage**

Moves the page.

**FRDocViewProcAction**

Performs a specified action.

**FRDocViewReleaseMachinePort**

Releases a [FR\\_WinPort](#) that acquired form page view using [FRDocViewGetMachinePort\(\)](#).

**FRDocViewScrollTo**

Scrolls the specified document view to location specified by *x* and *y* .

**FRDocViewSetLayoutMode**

Sets the page layout mode for a document view.

**FRDocViewSetRotation**

Sets the rotation factor.

**FRDocViewShowAnnots**

If *bShow* is set [TRUE](#) , the annotations will be shown. Otherwise not. This interface is not available in version 1.0.

**FRDocViewZoomTo**

Sets the zoom factor and zoom type to specified page view.

## Functions detail

### [FRDocViewCountPageViews](#)

**Syntax**

```
FS_INT32 FRDocViewCountPageViews (
    FR\_DocView docView
);
```

**Description**

Gets the number of page views for the specified document view.

**Parameter**

---

docView	[In] The document view whose page view count is obtained.
---------	---

---

**Return**

The number of [FR\\_PageView](#) object associated with the document view.

**Head file reference**

fr\_viewTempl.h: 34

**Related method**

[FRDocViewGetPageView](#)

[FRDocViewCountVisiblePageViews](#)

**FRDocViewCountVisiblePageViews****Syntax**

```
FS_INT32 FRDocViewCountVisiblePageViews (
    FR\_DocView docView
);
```

**Description**

Gets the number of visible page views for specified document view.

**Parameter**

---

docView	[In] The document view whose page view count is obtained.
---------	---

---

**Return**

The number of visible page views.

**Head file reference**

fr\_viewTempl.h: 40

**Related method**

[FRDocViewCountPageViews](#)

**FRDocViewDoPopUpMenu**

**Syntax**

```
FR_MenuItem FRDocViewDoPopUpMenu (
    FR\_DocView docView,
    FR\_Menu menu,
    FS\_INT32 xHit,
    FS\_INT32 yHit
);
```

**Description**

Displays the given menu as a pop-up menu anchored at *xHit* and *yHit*, which are in device space coordinates relative to *docView*.

**Parameter**

---

docView	[In] The document view in which the menu appears.
---------	---

---

menu	[In] The displayed menu.
------	--------------------------

---

xHit	[In] The x-coordinate of the upper left corner of the menu.
------	---

---

yHit	[In] The y-coordinate of the upper left corner of the menu.
------	---

---

**Return**

The menu item clicked by user.

**Head file reference**

fr\_viewTempl.h: 326

**Related method**

[FRMenuTrackPopup](#)

**FRDocViewDrawNow****Syntax**

```
void FRDocViewDrawNow (
    FR\_DocView docView
);
```

**Description**

Forces specified document view to redraw.

**Parameter**

---

docView	[In] The document view to redraw.
---------	-----------------------------------

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 289

**Related method**

[FRDocViewInvalidateRect](#)

**FRDocViewDrawRect****Syntax**

```
void FRDocViewDrawRect (
    FR\_DocView docView,
    FS\_Rect rect,
    FS\_ARGB fill_argb,
    FS\_BOOL bCompose
);
```

**Description**

Draws rectangle in the doc view.

**Parameter**

---

docView	[In] The input doc view.
---------	--------------------------

---

rect	[In] The input rectangle.
------	---------------------------

---

fill_argb	[In] The color to fill.
-----------	-------------------------

---

bCompose	[In] Whether to compose or not.
----------	---------------------------------

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 310

**Related method**

[FRDocViewDrawLine](#)

[FRDocViewDrawRectOutline](#)

[FRDocViewDrawPolygon](#)

[FRDocViewDrawPolygonOutline](#)

**FRDocViewGetCurrentPageView****Syntax**

```
FR_PageView FRDocViewGetCurrentPageView (
    FR\_DocView docView
);
```

**Description**

Gets current page view that is visible in screen.

**Parameter**

---

docView	[In] The document.
---------	--------------------

---

**Return**

Current [FR\\_PageView](#) that is visible in screen.

**Head file reference**

fr\_viewTempl.h: 54

**Related method**

[FRDocViewGetPageView](#)  
[FRDocViewGetPageViewAtPoint](#)  
[FRDocViewGetVisiblePageView](#)

**Note:** If more than one page may be visible. use [FRDocViewCountVisiblePageViews\(\)](#) and [FRDocViewGetVisiblePageView\(\)](#) instead.

**FRDocViewGetCurrentSnapshot****Syntax**

```
FS_DIBitmap FRDocViewGetCurrentSnapshot (
    FR\_DocView docView
);
```

**Description**

Gets current snapshot image that generated by using snapshot tool.

**Parameter**

---

docView	[In] The document view whose snapshot image are obtained.
---------	---

---

**Return**

A [FS\\_DIBitmap](#) object indicates current snapshot image. This [FS\\_DIBitmap](#) object must be destroyed by calling [FSDIBitmapDestroy\(\)](#).

**Head file reference**

fr\_viewTempl.h: 372

**Since**

## [SDK\\_LATEEST\\_VERSION > 1.0](#)

### **FRDocViewGetDocument**

#### **Syntax**

```
FR_Document FRDocViewGetDocument (
    FR\_DocView docView
);
```

#### **Description**

Gets the [FR\\_Document](#) associated with specified document view..

#### **Parameter**

---

docView	[In] The document view whose <a href="#">FR_Document</a> is obtained.
---------	---

---

#### **Return**

The [FR\\_Document](#) associated with the document view.

#### **Head file reference**

fr\_viewTempl.h: 21

#### **Related method**

[FRDocGetDocView](#)  
[FRDocGetCurrentDocView](#)  
[FRDocViewGetPageView](#)  
[FRPageViewGetDocument](#)

### **FRDocViewGetLayoutMode**

#### **Syntax**

```
FRDOCVIEW_LAYOUTMODE FRDocViewGetLayoutMode (
    FR\_DocView docView
);
```

#### **Description**

Gets the page layout mode.

#### **Parameter**

---

docView	[In] The document view whose layout mode is obtained.
---------	---

---

#### **Return**

The current page layout mode, a [FRDOC\\_LAYOUTMODE](#) enumeration value.

#### **Head file reference**

---

fr\_viewTempl.h: 170

## FRDocViewGetLRCViewText

### Syntax

```
void FRDocViewGetLRCViewText (
    FR\_DocView docView,
    FS\_WideString* outText,
    FS\_INT32 nCount
);
```

### Description

Gets the text of a page in LR Order.

### Parameter

docView	[In] The document view.
outText	[Out] It receives the text got.
nCount	[In] The count of page text to get, -1 for the end of the page.

### Return

void

### Head file reference

fr\_viewTempl.h: 444

### Since

[SDK\\_LATEEST\\_VERSION > 8.3.2](#)

## FRDocViewGetMachinePort

### Syntax

```
FR_WinPort FRDocViewGetMachinePort (
    FR\_DocView docView
);
```

### Description

Gets the platform-specific object needed to draw into page view using a platform's native graphic calls. In Win32 OS, it is a [FR\\_WinPort](#) object which contain the *HWND* handler and the device context. When done, release it using [FRDocViewReleaseMachinePort](#) ().

### Parameter

---

docView	[In] The document view whose <i>HWND</i> handler and device context are obtained.
---------	---

---

**Return**

The [FR\\_WinPort](#) object.

**Head file reference**

fr\_viewTempl.h: 340

**Related method**

[FRDocViewReleaseMachinePort](#)

**FRDocViewGetMaxScrollingSize****Syntax**

```
void FRDocViewGetMaxScrollingSize (
    FR\_DocView docView,
    FS\_INT32* outWidth,
    FS\_INT32* outHeight
);
```

**Description**

Gets the maximum scrolling size of the document view.

**Parameter**


---

docView	[In] The input document view.
---------	-------------------------------

---

outWidth	[Out] It receives the horizontal maximum scrolling size.
----------	--

---

outHeight	[Out] It receives the vertical maximum scrolling size.
-----------	--

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 278

**FRDocViewGetOCContext****Syntax**

```
FPD_OCContext FRDocViewGetOCContext (
    FR\_DocView docView
);
```

**Description**

Gets the *OCG* context of Reader doc view.

#### Parameter

---

docView	[In] The document view whose <i>ocg</i> context are obtained.
---------	---

---

#### Return

The *OCG* context of Reader doc view.

#### Head file reference

fr\_viewTempl.h: 363

## FRDocViewGetPageView

#### Syntax

```
FR_PageView FRDocViewGetPageView (
    FR\_DocView docView,
    FS\_INT32 iPage
);
```

#### Description

Gets the specified [FR\\_PageView](#) for specified document view.

#### Parameter

---

docView	[In] The document view whose <a href="#">FR_PageView</a> is obtained.
---------	---

---

iPage	[In] The index of page view to obtain. The index range is 0 to ( <a href="#">FRDocViewCountPageViews</a> ()-1).
-------	---

---

#### Return

The specified [FR\\_PageView](#) of the document view.

#### Head file reference

fr\_viewTempl.h: 28

#### Related method

[FRDocViewCountPageViews](#)  
[FRDocViewGetVisiblePageView](#)  
[FRDocViewGetPageViewAtPoint](#)  
[FRDocViewGetCurrentPageView](#)

## FRDocViewGetPageViewAtPoint

#### Syntax

```
FR_PageView FRDocViewGetPageViewAtPoint (
    FR\_DocView docView,
```

```
FS_DevicePoint point  
);
```

**Description**

Gets a specified [FR\\_PageView](#) with a point which is underlying to the area of page view.

**Parameter**

docView	[In] The document view whose <a href="#">FR_PageView</a> is obtained.
point	[In] The point of screen coordinate system to obtain a page view.

**Return**

The specified [FR\\_PageView](#) of the document view if the point is in the area of the page view, otherwise [NULL](#) .

**Head file reference**

fr\_viewTempl.h: 53

**Related method**

[FRDocViewGetPageView](#)  
[FRDocViewGetVisiblePageView](#)  
[FRDocViewGetCurrentPageView](#)

**FRDocViewGetRotation****Syntax**

```
FS_INT32 FRDocViewGetRotation (  
    FR\_DocView docView  
);
```

**Description**

Gets the rotation factor.

**Parameter**

docView	[In] The document view to obtain rotation factor.
---------	---

**Return**

A integer described in group [FR\\_RotationFlags](#) .

**Head file reference**

fr\_viewTempl.h: 149

**Related method**

## FRDocViewGetTagDocViewText

### Syntax

```
void FRDocViewGetTagDocViewText (
    FR\_DocView docView,
    FS\_WideString* outText,
    FS\_INT32 nCount
);
```

### Description

Gets the text of a page in Tag Order.

### Parameter

---

docView	[In] The document view.
---------	-------------------------

---

outText	[Out] It receives the text got.
---------	---------------------------------

---

nCount	[In] The count of page text to get, -1 for the end of the page.
--------	---

---

### Return

void

### Head file reference

fr\_viewTempl.h: 432

### Since

[SDK\\_LATEEST\\_VERSION > 8.3.2](#)

## FRDocViewGetThumbnailView

### Syntax

```
FR_ThumbnailView FRDocViewGetThumbnailView (
    FR\_DocView docView
);
```

### Description

Gets the thumbnail view related to the input document view.

### Parameter

---

docView	[In] The input document view.
---------	-------------------------------

---

### Return

A [FR\\_ThumbnailView](#) object related to the input document view.

**Head file reference**

fr\_viewTempl.h: 383

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FRDocViewGetVisiblePageView****Syntax**

```
FR_PageView FRDocViewGetVisiblePageView (
    FR\_DocView docView,
    FS\_INT32 iPage
);
```

**Description**

Gets the specified visible page.

**Parameter**

docView	[In] The document view whose page view is obtained.
iPage	[In] The index of the visible page. The index range is 0 to ( <a href="#">FRDocViewCountVisiblePageViews</a> ()-1).

**Return**

The specified [FR\\_PageView](#) of the document view.

**Head file reference**

fr\_viewTempl.h: 52

**Related method**

[FRDocViewGetPageView](#)  
[FRDocViewGetPageViewAtPoint](#)  
[FRDocViewGetCurrentPageView](#)

**FRDocViewGetZoom****Syntax**

```
FS_FLOAT FRDocViewGetZoom (
    FR\_DocView docView
);
```

**Description**

Gets the current zoom for page view. The zoom factor is point-to-point, not point-to-pixel.

**Parameter**

---

docView	[In] The document view.
---------	-------------------------

---

**Return**

The current zoom for page view.

**Head file reference**

fr\_viewTempl.h: 189

**Related method**

[FRDocViewGetZoomType](#)

[FRDocViewZoomTo](#)

**Note:** Current zoom, as a fixed number measured in units in which 1.0 is 100% zoom.

**FRDocViewGetZoomType****Syntax**

```
FRDOCVIEW_ZOOMTYPE FRDocViewGetZoomType (
    FR\_DocView docView
);
```

**Description**

Gets the current zoom type.

**Parameter**

---

docView	[In] The document view.
---------	-------------------------

---

**Return**

Current zoom type.

**Head file reference**

fr\_viewTempl.h: 194

**Related method**

[FRDocViewGetZoom](#)

[FRDocViewZoomTo](#)

**FRDocViewGoBack****Syntax**

```
void FRDocViewGoBack (
    FR\_DocView docView
);
```

**Description**

Goes to the previous view on the view stack, if a previous view exist.

**Parameter**

---

docView	[In] The document view.
---------	-------------------------

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 226

**Related method**

[FRDocViewGoForward](#)

**FRDocViewGoForward****Syntax**

```
void FRDocViewGoForward (
    FR\_DocView docView
);
```

**Description**

Goes to the next view on the view stack, if a next view exist.

**Parameter**

---

docView	[In] The document view.
---------	-------------------------

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 231

**Related method**

[FRDocViewGoBack](#)

**FRDocViewGotoPageView****Syntax**

```
void FRDocViewGotoPageView (
    FR\_DocView docView,
    FS\_INT32 iPage
);
```

**Description**

Goes to specified page, retaining the current display mode. It invalidates the display, but not always perform an immediately redraw.

### Parameter

---

docView	[In] The document view whose page view to go to.
iPage	[In] The specified page index. The index range is 0 to ( <a href="#">FRDocViewCountPageViews</a> ()-1).

---

### Return

void

#### Head file reference

fr\_viewTempl.h: 96

#### Related method

[FRDocViewGotoPageViewByPoint](#)  
[FRDocViewGotoPageViewByRect](#)

## FRDocViewGotoPageViewByPoint

### Syntax

```
void FRDocViewGotoPageViewByPoint (
    FR\_DocView docView,
    FS\_INT32 iPage,
    FS\_FLOAT left,
    FS\_FLOAT top
);
```

### Description

Goes to specified page and scroll page view to the location specified by *left* and *top* , retaining the current display mode. It invalidates the display, but not always perform an immediately redraw.

### Parameter

---

docView	[In] The document view whose page view to go to.
iPage	[In] The specified page index. The index range is 0 to ( <a href="#">FRDocViewCountPageViews</a> ()-1).
left	[In] The x-coordinate to scroll to. Specified in PDF space coordinates.

---

---

top	[In] The y-coordinate to scroll to. Specified in PDF space coordinates.
-----	---

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 103

**Related method**[FRDocViewGotoPageView](#)[FRDocViewGotoPageViewByRect](#)**FRDocViewGotoPageViewByRect****Syntax**

```
void FRDocViewGotoPageViewByRect (
    FR\_DocView docView,
    FS\_INT32 iPage,
    FS\_FloatRect rect
);
```

**Description**

Goes to specified page view and scroll to center of the rectangle. It always change the zoom level if necessary.

**Parameter**


---

docView	[In] The document view whose page view to go to.
---------	--

---

iPage	[In] The index of the visible page. The index range is 0 to ( <a href="#">FRDocViewCountVisiblePageViews</a> ()-1).
-------	---

---

rect	[In] The rectangle that is completely visible.
------	--

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 104

**Related method**[FRDocViewGotoPageView](#)[FRDocViewGotoPageViewByPoint](#)**FRDocViewGotoPageViewState**

**Syntax**

```
void FRDocViewGotoPageViewByState (
    FR\_DocView docView,
    FS\_INT32 iPage,
    FS\_INT32 iFitType,
    FS\_FLOAT* destArray,
    FS\_INT32 destArrayCount
);
```

**Description**

Goes to the page view by state.

**Parameter**

docView	[In] The document view whose page view to go to.
iPage	[In] The specified page index. The index range is 0 to ( <a href="#">FRDocViewCountPageViews</a> ()-1).
iFitType	[In] The input fit type.
destArray	[In] The input destination array.
destArrayCount	[In] The count of the destination array.

**Return**

void

**Head file reference**

[fr\\_viewTempl.h](#): 393

**Related method**

[FRDocViewGotoPageViewByPoint](#)

[FRDocViewGotoPageViewByRect](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRDocViewInvalidateRect****Syntax**

```
void FRDocViewInvalidateRect (
    FR\_DocView docView,
    FS\_Rect rect
);
```

**Description**

Redraws the specified area of document view immediately.

**Parameter**

---

docView	[In] The document view in which a region is invalidated.
rect	[In] The rectangle to invalidate, specified in device space coordinates.

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 294

**Related method**

[FRDocViewDrawNow](#)

**FRDocViewIsValidPageView****Syntax**

```
FS_BOOL FRDocViewIsValidPageView (
    FR\_DocView docView,
    FR\_PageView pPageView
);
```

**Description**

Checks whether the page view is valid or not.

**Parameter**

---

docView	[In] The document view.
pPageView	[In] The page view.

---

**Return**

TRUE if the page view is valid, otherwise FALSE.

**Head file reference**

fr\_viewTempl.h: 421

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

**FRDocViewMovePage**

**Syntax**

```
void FRDocViewMovePage (
    FR\_DocView docView,
    FS\_DevicePoint ptDest,
    FS\_DevicePoint ptSrc
);
```

**Description**

Moves the page.

**Parameter**

---

docView	[In] The document view.
---------	-------------------------

---

ptDest	[In] The dest point.
--------	----------------------

---

ptSrc	[In] The source point.
-------	------------------------

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 409

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

**FRDocViewProcAction****Syntax**

```
void FRDocViewProcAction (
    FR\_DocView docView,
    FPD\_Action action
);
```

**Description**

Performs a specified action.

**Parameter**

---

docView	[In] The document view.
---------	-------------------------

---

action	[In] A <a href="#">FPD_Action</a> object to be performed.
--------	---

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 257

**FRDocViewReleaseMachinePort****Syntax**

```
FS_BOOL FRDocViewReleaseMachinePort (
    FR\_DocView docView,
    FR\_WinPort port
);
```

**Description**

Releases a [FR\\_WinPort](#) that acquired from page view using [FRDocViewGetMachinePort](#) ().

**Parameter**

---

docView	[In] The document view whose <i>HWND</i> handler and device context are released.
port	[In] The platform-dependent object to release.

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_viewTempl.h: 343

**Related method**[FRDocViewGetMachinePort](#)**FRDocViewScrollTo****Syntax**

```
void FRDocViewScrollTo (
    FR\_DocView docView,
    FS\_INT32 x,
    FS\_INT32 y
);
```

**Description**

Scrolls the specified document view to location specified by *x* and *y*.

**Parameter**

---

docView	[In] The document view to scroll.
---------	-----------------------------------

---

x	[In] The x-coordinate to scroll to, specified in device space coordinates.
---	--

y	[In] The y-coordinate to scroll to, specified in device space coordinates.
---	--

**Return**

void

**Head file reference**

fr\_viewTempl.h: 267

**FRDocViewSetLayoutMode****Syntax**

```
void FRDocViewSetLayoutMode (
    FR\_DocView docView,
    FRDOCVIEW\_LAYOUTMODE mode
);
```

**Description**

Sets the page layout mode for a document view.

**Parameter**

docView	[In] The document view whose layout mode is set.
---------	--

mode	[In] The new layout mode for document view.
------	---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 179

**FRDocViewSetRotation****Syntax**

```
void FRDocViewSetRotation (
    FR\_DocView docView,
    FS\_INT32 nFlag
);
```

**Description**

Sets the rotation factor.

**Parameter**


---

docView	[In] The document view to obtain rotation factor.
---------	---

---

nFlag	[In] A integer described in group <a href="#">FR_RotationFlags</a> .
-------	--

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 159

**Related method****FRDocViewShowAnnots****Syntax**

```
void FRDocViewShowAnnots (
    FR\_DocView docView,
    FS\_PtrArray* annots,
    FS\_BOOL bShow
);
```

**Description**

If *bShow* is set [TRUE](#) , the annotations will be shown. Otherwise not. This interface is not available in version 1.0.

**Parameter**


---

docView	[In] The document view.
---------	-------------------------

---

annots	[In] The array of annotations to be shown.
--------	--

---

bShow	[In] Whether to show or not.
-------	------------------------------

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 246

**FRDocViewZoomTo****Syntax**

```
void FRDocViewZoomTo (
    FR\_DocView docView,
```

---

[FRDOCVIEW\\_ZOOMTYPE](#) mode,  
[FS\\_FLOAT](#) scale  
);

**Description**

Sets the zoom factor and zoom type to specified page view.

**Parameter**


---

docView	[In] The document view.
mode	[In] The zoom mode to set.
scale	[In] The zoom factor, specified as a magnification factor(for example, 1.0 displays the document at actual size). This is ignore unless iFittype parameter is <a href="#"><u>FR_ZOOM_MODE_ACTUAL_SCALE</u></a> or <a href="#"><u>FR_ZOOM_MODE_NONE</u></a> .

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 195

**Related method**

[FRDocViewGetZoom](#)

[FRDocViewGetZoomType](#)

## FR\_Edit

**Description****Definitions****Definitions summary**

[FR\\_EMBED\\_CHARSET](#)  
[FR\\_VT\\_SECTION\\_FLG](#)  
[FR\\_VT\\_SOFTRETURN](#)  
[FR\\_VT\\_TEXT\\_HIDCHAR](#)  
[FR\\_VT\\_TEXT\\_UNICODEINVALID](#)  
[FR\\_VT\\_WORD\\_STYLE\\_BOLD](#)  
[FR\\_VT\\_WORD\\_STYLE\\_ITALIC](#)

**Definitions detail**

## FR\_EMBED\_CHARSET

**Syntax**

```
#define FR_EMBED_CHARSET 0x7FFFFFFF
```

**Description****Group**

[EmbedCharSet](#)

**Head file reference**

fr\_appExpT.h: 6965

## FR\_VT\_SECTION\_FLG

**Syntax**

```
#define FR_VT_SECTION_FLG 0x0D
```

**Description****Group**

[SpecificChars](#)

**Head file reference**

fr\_appExpT.h: 6950

## FR\_VT\_SOFTRETURN

**Syntax**

```
#define FR_VT_SOFTRETURN 0xFFFFE
```

**Description****Group**

[SpecificChars](#)

**Head file reference**

fr\_appExpT.h: 6948

## FR\_VT\_TEXT\_HIDCHAR

**Syntax**

```
#define FR_VT_TEXT_HIDCHAR 0xFFFF8
```

**Description****Group**

[SpecificChars](#)

**Head file reference**

fr\_appExpT.h: 6952

## FR\_VT\_TEXT\_UNICODEINVALID

**Syntax**

```
#define FR_VT_TEXT_UNICODEINVALID 0xFFFF
```

**Description**

**Group**

[SpecificChars](#)

**Head file reference**

fr\_appExpT.h: 6954

## FR\_VT\_WORD\_STYLE\_BOLD

**Syntax**

```
#define FR_VT_WORD_STYLE_BOLD 0x40000
```

**Description**

**Group**

[VTWordStyle](#)

**Head file reference**

fr\_appExpT.h: 6976

## FR\_VT\_WORD\_STYLE\_ITALIC

**Syntax**

```
#define FR_VT_WORD_STYLE_ITALIC 0x40
```

**Description**

**Group**

[VTWordStyle](#)

**Head file reference**

fr\_appExpT.h: 6977

## Callbacks

### Callbacks summary

## Callbacks detail

### Structures

#### Structures summary

##### [FR\\_EDIT\\_FontData](#)

#### Structs detail

##### [FR\\_EDIT\\_FontData](#)

###### Syntax

```
typedef struct __FR_EDIT_FontData__{
    FPD\_Font fpdFont,
    FS\_ByteString fontAName
}FR_EDIT_FontData;
```

###### Description

###### Head file reference

fr\_appExpT.h: 6988

**fpdFont**

**fontAName**

## [FR\\_Edit\\_FontMap](#)

### Description

### Enumerations

#### Enumerations summary

##### [FRFMTribool](#)

#### Enumerations detail

##### [FRFMTribool](#)

###### Syntax

```
enum FRFMTribool{
    FR\_FMFalse,
    FR\_FMTrue,
    FR\_FMIIndeterminate
};
```

###### Description

**Head file reference**

fr\_appExpT.h: 6802

**FR\_FMFalse****FR\_FMTrue****FR\_FMIIndeterminate**

## **FR\_Edit\_Iterator**

### Description

## **FR>Edit\_UndoItem**

### Description

## **FR\_EmptyFrameWndViewEventHandler**

### [Return from Used by](#)

### Description

The [FR\\_EmptyFrameWndViewEventHandler](#) object represents the event handler of the empty frame window view. The plug-in can register the event handler to process the event when the plug-in creates the empty frame window. See  
[FRAppCreateEmptyFrameViewEventHandler](#) and [FRAppCreateAnEmptyFrameWnd2](#).

### Returned from

### [FRAppCreateEmptyFrameViewEventHandler](#)

### Used by

[FRAppCreateAnEmptyFrameWnd2](#)  
[FRAppCreateEmptyFrameViewEventHandler](#)  
[FRAppDestroyEmptyFrameViewEventHandler](#)

## Callbacks

### Callbacks summary

#### [FREmptyFrameViewOnActivate](#)

A callback for the event handler of the empty frame window view.

**FREmptyFrameViewOnDeactivate**

A callback for the event handler of the empty frame window view.

**FREmptyFrameViewOnViewSize**

A callback for the event handler of the empty frame window view.

**FREmptyFrameViewOnSetFocus**

A callback for the event handler of the empty frame window view.

**FREmptyFrameViewOnWillClose**

A callback for the event handler of the empty frame window view.

**FREmptyFrameViewOnViewCreate**

A callback for the event handler of the empty frame window view.

**FREmptyFrameViewOnDidClose**

A callback for the event handler of the empty frame window view.

**FREmptyFrameViewOnWillDetachChildFrame**

A callback for the event handler of the childFrame window detach from mainframe.

**FREmptyFrameViewOnActivate2**

A callback for the event handler of the empty frame window view.

**FREmptyFrameViewOnDeactivate2**

A callback for the event handler of the empty frame window view.

**FREmptyFrameViewGetDisplayViewHWnd**

A callback for the event handler to get display view ptr.

## Callbacks detail

### FREmptyFrameViewOnActivate

#### Syntax

```
typedef void (*FREmptyFrameViewOnActivate)(  
    FS_LVOID clientData,  
    HWND hView  
)
```

#### Description

A callback for the event handler of the empty frame window view. It is called to notify the plug-in when the view is activated.

#### Parameter

clientData	[In] The user-supplied data.
------------	------------------------------

hView	[In] The empty frame window view.
-------	-----------------------------------

#### Return

void.

#### Head file reference

fr\_appExpT.h: 6499

#### Group

[FR\\_EmptyFrameWndViewEventHandlerCallbacksRec](#)

**Since**

[SDK LATEST VERSION > 7.3](#)

**FREmptyFrameViewOnDeactivate****Syntax**

```
typedef void (*FREmptyFrameViewOnDeactivate)(  
    FS\_LPVVOID clientData,  
    HWND hView  
);
```

**Description**

A callback for the event handler of the empty frame window view. It is called to notify the plug-in when the view is deactivated.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

hView	[In] The empty frame window view.
-------	-----------------------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 6514

**Group**

[FR\\_EmptyFrameWndViewEventHandlerCallbacksRec](#)

**Since**

[SDK LATEST VERSION > 7.3](#)

**FREmptyFrameViewOnViewSize****Syntax**

```
typedef void (*FREmptyFrameViewOnViewSize)(  
    FS\_LPVVOID clientData,  
    HWND hView,  
    FS\_Rect rtClient  
);
```

**Description**

A callback for the event handler of the empty frame window view. It is called to notify the plug-in when the size of the view is changed.

**Parameter**


---

clientData	[In] The user-supplied data.
hView	[In] The empty frame window view.
rtClient	[In] The rectangle area of the view.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 6530

**Group**[FR\\_EmptyFrameWndViewEventHandlerCallbacksRec](#)**Since**[SDK LATEEST VERSION > 7.3](#)**FREmptyFrameViewOnSetFocus****Syntax**

```
typedef void (*FREmptyFrameViewOnSetFocus)(  
    FS\_LPVVOID clientData,  
    HWND hView,  
    HWND hOldWnd  
)
```

**Description**

A callback for the event handler of the empty frame window view. It is called to notify the plug-in when the view is focused.

**Parameter**


---

clientData	[In] The user-supplied data.
hView	[In] The empty frame window view.
hOldWnd	[In] The view of the old one.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 6546

**Group**

[FR\\_EmptyFrameWndViewEventHandlerCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FREmptyFrameViewOnWillClose****Syntax**

```
typedef void (*FREmptyFrameViewOnWillClose)(  
    FS\_LPVOID clientData,  
    HWND hView  
)
```

**Description**

A callback for the event handler of the empty frame window view. It is called to notify the plug-in when the view will be closed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

hView	[In] The empty frame window view.
-------	-----------------------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 6561

**Group**

[FR\\_EmptyFrameWndViewEventHandlerCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FREmptyFrameViewOnViewCreate****Syntax**

```
typedef void (*FREmptyFrameViewOnViewCreate)(  
    FS\_LPVOID clientData,  
    HWND hView  
)
```

**Description**

A callback for the event handler of the empty frame window view. It is called to notify the plug-in when the view is created.

**Parameter**

---

clientData	[In] The user-supplied data.
hView	[In] The empty frame window view.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 6576

**Group**

[FR\\_EmptyFrameWndViewEventHandlerCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FREmptyFrameViewOnDidClose****Syntax**

```
typedef void (*FREmptyFrameViewOnDidClose)(  
    FS\_LPVOID clientData,  
    HWND hFrameWnd  
)
```

**Description**

A callback for the event handler of the empty frame window view. It is called to notify the plug-in when the view had been closed.

**Parameter**

---

clientData	[In] The user-supplied data.
hFrameWnd	[In] The empty frame window.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 6590

**Group**

[FR\\_EmptyFrameWndViewEventHandlerCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 9.0](#)

**FREmptyFrameViewOnWillDetachChildFrame****Syntax**

```
typedef void (*FREmptyFrameViewOnWillDetachChildFrame)(  
    FS\_LPVVOID clientData,  
    HWND hFrameWnd  
);
```

**Description**

A callback for the event handler of the childFrame window detach from mainframe. It is called to notify the plug-in when the view will been detach.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

hFrameWnd	[In] The empty frame window.
-----------	------------------------------

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 6604

**Group**

[FR\\_EmptyFrameWndViewEventHandlerCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 9.4](#)

**FREmptyFrameViewOnActivate2****Syntax**

```
typedef void (*FREmptyFrameViewOnActivate2)(  
    FS\_LPVVOID clientData,  
    HWND hView,  
    HWND hMainframe  
);
```

**Description**

A callback for the event handler of the empty frame window view. It is called to notify the plug-in when the view is activated.

**Parameter**


---

clientData	[In] The user-supplied data.
hView	[In] The empty frame window view.
hMainframe	[In] The empty frame window mainframe.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 6620

**Group**[FR\\_EmptyFrameWndViewEventHandlerCallbacksRec](#)**Since**[SDK LATEEST VERSION > 9.4](#)**FREmptyFrameViewOnDeactivate2****Syntax**

```
typedef void (*FREmptyFrameViewOnDeactivate2)(
    FS\_LPVVOID clientData,
    HWND hView,
    HWND hMainframe
);
```

**Description**

A callback for the event handler of the empty frame window view. It is called to notify the plug-in when the view is deactivated.

**Parameter**


---

clientData	[In] The user-supplied data.
hView	[In] The empty frame window view.
hMainframe	[In] The empty frame window mainframe.

---

**Return**

void.

**Head file reference**

---

fr\_appExpT.h: 6636

**Group**[FR\\_EmptyFrameWndViewEventHandlerCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 9.4](#)**FREmptyFrameViewGetDisplayViewHWnd****Syntax**

```
typedef HWND (*FREmptyFrameViewGetDisplayViewHWnd)(
    FS_LPVVOID clientData,
    HWND hPluginView
);
```

**Description**

A callback for the event handler to get display view ptr. It is called to get display view from plug-in.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

hPluginView	[In] The window handle of pluginview.
-------------	---------------------------------------

---

**Return**

HWND The display view handle.

**Head file reference**

fr\_appExpT.h: 6651

**Group**[FR\\_EmptyFrameWndViewEventHandlerCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 9.4](#)**FR\_FormatTools**[Return from Used by](#)**Description**

The format tool is used to set the format of the PDF object and text. You can set the format like font name, font size, color and so on.

Returned from

[FRFormatToolsGet](#)

Used by

[FRFormatToolsActivateFormatCategory](#)  
[FRFormatToolsCleanOwnerFontNameArr](#)  
[FRFormatToolsCountFontList](#)  
[FRFormatToolsEnable](#)  
[FRFormatToolsEnableButton](#)  
[FRFormatToolsFindFontName](#)  
[FRFormatToolsGetAlign](#)  
[FRFormatToolsGetCharHorzScale](#)  
[FRFormatToolsGetCharSpace](#)  
[FRFormatToolsGetCross](#)  
[FRFormatToolsGetFillColor](#)  
[FRFormatToolsGetFontFaceName](#)  
[FRFormatToolsGetFontListItem](#)  
[FRFormatToolsGetFontName](#)  
[FRFormatToolsGetFontSize](#)  
[FRFormatToolsGetLineColor](#)  
[FRFormatToolsGetLineLeading](#)  
[FRFormatToolsGetSubScript](#)  
[FRFormatToolsGetSuperScript](#)  
[FRFormatToolsGetTextColor](#)  
[FRFormatToolsGetUnderline](#)  
[FRFormatToolsGetWordSpace](#)  
[FRFormatToolsGetWritingDirection](#)  
[FRFormatToolsHideButton](#)  
[FRFormatToolsIsBold](#)  
[FRFormatToolsIsButtonEnabled](#)  
[FRFormatToolsIsButtonVisible](#)  
[FRFormatToolsIsEnabled](#)  
[FRFormatToolsIsItalic](#)  
[FRFormatToolsIsVisible](#)  
[FRFormatToolsSetAlign](#)  
[FRFormatToolsSetBold](#)  
[FRFormatToolsSetCharHorzScale](#)  
[FRFormatToolsSetCharSpace](#)  
[FRFormatToolsSetCross](#)  
[FRFormatToolsSetEvent](#)  
[FRFormatToolsSetFillColor](#)  
[FRFormatToolsSetFontName](#)  
[FRFormatToolsSetFontNameFirst](#)  
[FRFormatToolsSetFontSize](#)  
[FRFormatToolsSetFontSizeFirst](#)  
[FRFormatToolsSetFormatContextTitle](#)  
[FRFormatToolsSetItalic](#)  
[FRFormatToolsSetLineColor](#)  
[FRFormatToolsSetLineLeading](#)  
[FRFormatToolsSetOwnerFontNameArr](#)  
[FRFormatToolsSetSubScript](#)  
[FRFormatToolsSetSuperScript](#)  
[FRFormatToolsSetTextColor](#)  
[FRFormatToolsSetUnderline](#)

[\*\*FRFormatToolsSetWordSpace\*\*](#)  
[\*\*FRFormatToolsSetWritingDirection\*\*](#)  
[\*\*FRFormatToolsShow\*\*](#)

## Definitions

### Definitions summary

#### [\*\*FR\\_FMT\\_ALIGN\\_CENTER\*\*](#)

The ID of the center alignment setting button.

#### [\*\*FR\\_FMT\\_ALIGN\\_LEFT\*\*](#)

The ID of the left alignment setting button.

#### [\*\*FR\\_FMT\\_ALIGN\\_RIGHT\*\*](#)

The ID of the right alignment setting button.

#### [\*\*FR\\_FMT\\_BOLD\*\*](#)

The ID of the bold setting button.

#### [\*\*FR\\_FMT\\_BORDER\\_COLOR\*\*](#)

The ID of the border color setting button.

#### [\*\*FR\\_FMT\\_CHAR\\_SPACE\*\*](#)

The ID of the character space setting button.

#### [\*\*FR\\_FMT\\_CROSS\*\*](#)

The ID of the cross setting button.

#### [\*\*FR\\_FMT\\_DEDENT\*\*](#)

The ID of the dedent setting button.

#### [\*\*FR\\_FMT\\_FILL\\_COLOR\*\*](#)

The ID of the fill color setting button.

#### [\*\*FR\\_FMT\\_FONT\\_NAME\*\*](#)

The ID of the font name setting button.

#### [\*\*FR\\_FMT\\_FONT\\_SIZE\*\*](#)

The ID of the font size setting button.

#### [\*\*FR\\_FMT\\_HORZ\\_SCALE\*\*](#)

The ID of the horizon scale setting button.

#### [\*\*FR\\_FMT\\_INDENT\*\*](#)

The ID of the indent setting button.

#### [\*\*FR\\_FMT\\_ITALIC\*\*](#)

The ID of the italic setting button.

#### [\*\*FR\\_FMT\\_LINE\\_LEADING\*\*](#)

The ID of the line leading setting button.

#### [\*\*FR\\_FMT\\_LINECOLOR\*\*](#)

The ID of the line color setting button.

#### [\*\*FR\\_FMT\\_OPACITY\*\*](#)

The ID of the opacity setting button.

#### [\*\*FR\\_FMT\\_SUBSCRIPT\*\*](#)

The ID of the subscript setting button.

#### [\*\*FR\\_FMT\\_SUPERSCRIPT\*\*](#)

The ID of the superscript setting button.

#### [\*\*FR\\_FMT\\_TEXT\\_COLOR\*\*](#)

The ID of the color setting button.

#### [\*\*FR\\_FMT\\_UNDERLINE\*\*](#)

The ID of the underline setting button.

#### [\*\*FR\\_FMT\\_WORDSPACE\*\*](#)

The ID of the italic setting button.

#### **FR\_FMT\_WRITING\_DIR**

The ID of the writing direction setting button.

### **Definitions detail**

#### **FR\_FMT\_ALIGN\_CENTER**

##### **Syntax**

```
#define FR_FMT_ALIGN_CENTER 9
```

##### **Description**

The ID of the center alignment setting button.

##### **Group**

[FRFormatToolsButtonIDDefs](#)

##### **Head file reference**

fr\_barExpT.h: 781

#### **FR\_FMT\_ALIGN\_LEFT**

##### **Syntax**

```
#define FR_FMT_ALIGN_LEFT 8
```

##### **Description**

The ID of the left alignment setting button.

##### **Group**

[FRFormatToolsButtonIDDefs](#)

##### **Head file reference**

fr\_barExpT.h: 779

#### **FR\_FMT\_ALIGN\_RIGHT**

##### **Syntax**

```
#define FR_FMT_ALIGN_RIGHT 10
```

##### **Description**

The ID of the right alignment setting button.

##### **Group**

[FRFormatToolsButtonIDDefs](#)

##### **Head file reference**

fr\_barExpT.h: 783

## FR\_FMT\_BOLD

**Syntax**

```
#define FR_FMT_BOLD 6
```

**Description**

The ID of the bold setting button.

**Group**

[FRFormatToolsButtonIDDefs](#)

**Head file reference**

fr\_barExpT.h: 775

## FR\_FMT\_BORDER\_COLOR

**Syntax**

```
#define FR_FMT_BORDER_COLOR 4
```

**Description**

The ID of the border color setting button.

**Group**

[FRFormatToolsButtonIDDefs](#)

**Head file reference**

fr\_barExpT.h: 771

## FR\_FMT\_CHAR\_SPACE

**Syntax**

```
#define FR_FMT_CHAR_SPACE 11
```

**Description**

The ID of the character space setting button.

**Group**

[FRFormatToolsButtonIDDefs](#)

**Head file reference**

fr\_barExpT.h: 785

## FR\_FMT\_CROSS

**Syntax**

```
#define FR_FMT_CROSS 15
```

**Description**

The ID of the cross setting button.

**Group**

[FRFormatToolsButtonIDDefs](#)

**Head file reference**

fr\_barExpT.h: 793

**FR\_FMT\_DEDENT****Syntax**

```
#define FR_FMT_DEDENT 19
```

**Description**

The ID of the dedent setting button.

**Group**

[FRFormatToolsButtonIDDefs](#)

**Head file reference**

fr\_barExpT.h: 801

**FR\_FMT\_FILL\_COLOR****Syntax**

```
#define FR_FMT_FILL_COLOR 5
```

**Description**

The ID of the fill color setting button.

**Group**

[FRFormatToolsButtonIDDefs](#)

**Head file reference**

fr\_barExpT.h: 773

**FR\_FMT\_FONT\_NAME****Syntax**

```
#define FR_FMT_FONT_NAME 2
```

**Description**

The ID of the font name setting button.

**Group**

[FRFormatToolsButtonIDDefs](#)**Head file reference**

fr\_barExpT.h: 767

**FR\_FMT\_FONT\_SIZE****Syntax**

#define FR\_FMT\_FONT\_SIZE 3

**Description**

The ID of the font size setting button.

**Group**[FRFormatToolsButtonIDDefs](#)**Head file reference**

fr\_barExpT.h: 769

**FR\_FMT\_HORZ\_SCALE****Syntax**

#define FR\_FMT\_HORZ\_SCALE 12

**Description**

The ID of the horizon scale setting button.

**Group**[FRFormatToolsButtonIDDefs](#)**Head file reference**

fr\_barExpT.h: 787

**FR\_FMT\_INDENT****Syntax**

#define FR\_FMT\_INDENT 18

**Description**

The ID of the indent setting button.

**Group**[FRFormatToolsButtonIDDefs](#)**Head file reference**

fr\_barExpT.h: 799

## FR\_FMT\_ITALIC

### Syntax

```
#define FR_FMT_ITALIC 7
```

### Description

The ID of the italic setting button.

### Group

[FRFormatToolsButtonIDDefs](#)

### Head file reference

fr\_barExpT.h: 777

## FR\_FMT\_LINE\_LEADING

### Syntax

```
#define FR_FMT_LINE_LEADING 13
```

### Description

The ID of the line leading setting button.

### Group

[FRFormatToolsButtonIDDefs](#)

### Head file reference

fr\_barExpT.h: 789

## FR\_FMT\_LINECOLOR

### Syntax

```
#define FR_FMT_LINECOLOR 4
```

### Description

The ID of the line color setting button.

### Group

[FRFormatToolsButtonIDDefs](#)

### Head file reference

fr\_barExpT.h: 805

## FR\_FMT\_OPACITY

### Syntax

```
#define FR_FMT_OPACITY 22
```

**Description**

The ID of the opacity setting button.

**Group**

[FRFormatToolsButtonIDDefs](#)

**Head file reference**

fr\_barExpT.h: 809

## FR\_FMT\_SUBSCRIPT

**Syntax**

```
#define FR_FMT_SUBSCRIPT 17
```

**Description**

The ID of the subscript setting button.

**Group**

[FRFormatToolsButtonIDDefs](#)

**Head file reference**

fr\_barExpT.h: 797

## FR\_FMT\_SUPERSCRIPT

**Syntax**

```
#define FR_FMT_SUPERSCRIPT 16
```

**Description**

The ID of the superscript setting button.

**Group**

[FRFormatToolsButtonIDDefs](#)

**Head file reference**

fr\_barExpT.h: 795

## FR\_FMT\_TEXT\_COLOR

**Syntax**

```
#define FR_FMT_TEXT_COLOR 1
```

**Description**

The ID of the color setting button.

**Group**

[FRFormatToolsButtonIDDefs](#)**Head file reference**

fr\_barExpT.h: 765

**FR\_FMT\_UNDERLINE****Syntax**

#define FR\_FMT\_UNDERLINE 14

**Description**

The ID of the underline setting button.

**Group**[FRFormatToolsButtonIDDefs](#)**Head file reference**

fr\_barExpT.h: 791

**FR\_FMT\_WORDSPACE****Syntax**

#define FR\_FMT\_WORDSPACE 20

**Description**

The ID of the italic setting button.

**Group**[FRFormatToolsButtonIDDefs](#)**Head file reference**

fr\_barExpT.h: 803

**FR\_FMT\_WRITING\_DIR****Syntax**

#define FR\_FMT\_WRITING\_DIR 21

**Description**

The ID of the writing direction setting button.

**Group**[FRFormatToolsButtonIDDefs](#)**Head file reference**

fr\_barExpT.h: 807

## Enumerations

### Enumerations summary

#### [\*\*FRFormatToolArrangeAlignInfo\*\*](#)

The alignment type of arrange info.

#### [\*\*FRFormatToolArrangeCenterInfo\*\*](#)

The page centering type of arrange info.

#### [\*\*FRFormatToolArrangeDistributeInfo\*\*](#)

The distribution type of arrange info.

#### [\*\*FRFormatToolArrangeInfo\*\*](#)

The arrange info for selected annotations or form fields.

#### [\*\*FRFormatToolContextCategoryType\*\*](#)

The format category type.

#### [\*\*FRFormatToolWritingDirection\*\*](#)

The writing direction.

### Enumerations detail

#### [\*\*FRFormatToolArrangeAlignInfo\*\*](#)

##### **Syntax**

```
enum FRFormatToolArrangeAlignInfo{
    FRFormatToolArrangeAlignInfo\_LEFT,
    FRFormatToolArrangeAlignInfo\_RIGHT,
    FRFormatToolArrangeAlignInfo\_TOP,
    FRFormatToolArrangeAlignInfo\_BOTTOM,
    FRFormatToolArrangeAlignInfo\_VER,
    FRFormatToolArrangeAlignInfo\_HOR /\* Horizon alignment. \*/
};
```

##### **Description**

The alignment type of arrange info.

##### **Head file reference**

fr\_barExpT.h: 825

#### **FRFormatToolArrangeAlignInfo\_LEFT**

Left alignment.

#### **FRFormatToolArrangeAlignInfo\_RIGHT**

Right alignment.

#### **FRFormatToolArrangeAlignInfo\_TOP**

Top alignment.

#### **FRFormatToolArrangeAlignInfo\_BOTTOM**

Bottom alignment.

#### **FRFormatToolArrangeAlignInfo\_VER**

Vertical alignment.

**FRFormatToolArrangeAlignInfo\_HOR /\*\* Horizon alignment. \*/****FRFormatToolArrangeCenterInfo****Syntax**

```
enum FRFormatToolArrangeCenterInfo{
    FRFormatToolArrangeCenterInfo_VER,
    FRFormatToolArrangeCenterInfo_HOR,
    FRFormatToolArrangeCenterInfo_BOTH
};
```

**Description**

The page centering type of arrange info.

**Head file reference**

fr\_barExpT.h: 838

**FRFormatToolArrangeCenterInfo\_VER**

Vertical page centering.

**FRFormatToolArrangeCenterInfo\_HOR**

Horizon page centering.

**FRFormatToolArrangeCenterInfo\_BOTH**

Both vertical and horizon centering.

**FRFormatToolArrangeDistributeInfo****Syntax**

```
enum FRFormatToolArrangeDistributeInfo{
    FRFormatToolArrangeDistributeInfo_VER,
    FRFormatToolArrangeDistributeInfo_HOR /** @brief Horizon distribution. */
};
```

**Description**

The distribution type of arrange info.

**Head file reference**

fr\_barExpT.h: 848

**FRFormatToolArrangeDistributeInfo\_VER**

@brief Vertical distribution.

**FRFormatToolArrangeDistributeInfo\_HOR /\*\* @brief Horizon distribution. \*/****FRFormatToolArrangeInfo****Syntax**

```
enum FRFormatToolArrangeInfo{
    FRFormatToolArrangeInfo_ALIGN,
```

---

```
FRFormatToolArrangeInfo_CENTER,
FRFormatToolArrangeInfo_DISTRIBUTE /** The arrange info of distribution.
*/
};
```

**Description**

The arrange info for selected annotations or form fields.

**Head file reference**

fr\_barExpT.h: 815

**FRFormatToolArrangeInfo\_ALIGN**

The arrange info of alignment.

**FRFormatToolArrangeInfo\_CENTER**

The arrange info of page centering.

**FRFormatToolArrangeInfo\_DISTRIBUTE /\*\* The arrange info of distribution. \*/****FRFormatToolContextCategoryType****Syntax**

```
enum FRFormatToolContextCategoryType{
    FR_CATOGRY_COMMENT_FORMAT,
    FR_CATOGRY_OBJECT_FORMAT,
    FR_CATOGRY_ARRANGE_FORMAT
};
```

**Description**

The format category type.

**Head file reference**

fr\_barExpT.h: 866

**FR\_CATOGRY\_COMMENT\_FORMAT**

@brief The Comment format category type

**FR\_CATOGRY\_OBJECT\_FORMAT**

@brief The object format category type

**FR\_CATOGRY\_ARRANGE\_FORMAT**

@brief The arrange format category type

**FRFormatToolWritingDirection****Syntax**

```
enum FRFormatToolWritingDirection{
    FR_LEFT_TO_RIGHT,
    FR_RIGHT_TO_LEFT /** @brief The writing direction of right-to-left. */
};
```

**Description**

The writing direction.

**Head file reference**

fr\_barExpT.h: 857

**FR\_LEFT\_TO\_RIGHT**

@brief The writing direction of left-to-right.

**FR\_RIGHT\_TO\_LEFT /\*\* @brief The writing direction of right-to-left. \*/**

## Callbacks

### Callbacks summary

**FRFormatToolOnFontNameChanged**

It is called by Foxit Reader when the font name is changed.

**FRFormatToolOnFontSizeChanged**

It is called by Foxit Reader when the font size is changed.

**FRFormatToolOnTextColorChanged**

It is called by Foxit Reader when the text color is changed.

**FRFormatToolOnLineColorChanged**

It is called by Foxit Reader when the line color is changed.

**FRFormatToolOnFillColorChanged**

It is called by Foxit Reader when the filling color is changed.

**FRFormatToolOnBoldChanged**

It is called by Foxit Reader when the bold setting is changed.

**FRFormatToolOnItalicChanged**

It is called by Foxit Reader when the italic setting is changed.

**FRFormatToolOnAlignChanged**

It is called by Foxit Reader when the alignment setting is changed.

**FRFormatToolOnCharSpaceChanged**

It is called by Foxit Reader when the character space setting is changed.

**FRFormatToolOnCharHorzScaleChanged**

It is called by Foxit Reader when the character horizon scale setting is changed.

**FRFormatToolOnLineLeadingChanged**

It is called by Foxit Reader when the line leading setting is changed.

**FRFormatToolOnUnderlineChanged**

It is called by Foxit Reader when the underline setting is changed.

**FRFormatToolOnCrossChanged**

It is called by Foxit Reader when the cross setting is changed.

**FRFormatToolOnSuperScriptChanged**

It is called by Foxit Reader when the superscript setting is changed.

**FRFormatToolOnSubScriptChanged**

It is called by Foxit Reader when the subscript setting is changed.

**FRFormatToolOnInDentChanged**

It is called by Foxit Reader when the indent setting is changed.

**FRFormatToolOnDeDentChanged**

It is called by Foxit Reader when the dedent setting is changed.

**FRFormatToolOnWordSpaceChanged**

It is called by Foxit Reader when the word space setting is changed.

**FRFormatToolOnArrangeAlign**

It is called by Foxit Reader when the alignment of the arrange info is changed.

**FRFormatToolOnArrangeCenter**

It is called by Foxit Reader when the page centering of the arrange info is changed.

**FRFormatToolOnArrangeDistribute**

It is called by Foxit Reader when the distribution of the arrange info is changed.

**FRFormatToolGetArrangeEnable**

It is called by Foxit Reader to determine whether the arrange setting is enabled or not.

**FRFormatToolGetLineColorEnableNoColor**

It is called by Foxit Reader to determine whether the line color setting is enabled or not.

**FRFormatToolGetFillColorEnableNoColor**

It is called by Foxit Reader to determine whether the filling color setting is enabled or not.

**FRFormatToolOnWritingDirChanged**

It is called by Foxit Reader when the writing direction is changed.

**FRFormatToolOnOpacityChanged**

It is called by Foxit Reader when the opacity setting is changed.

**FRFormatToolOnOpacityScroll**

It is called by Foxit Reader when the opacity setting is scrolling.

**Callbacks detail****FRFormatToolOnFontNameChanged****Syntax**

```
typedef void (*FRFormatToolOnFontNameChanged)(  
    FS_LPVVOID clientData,  
    FS_LPCWSTR lpwsFontName,  
    FS_LPCWSTR lpwsScriptName  
>;
```

**Description**

It is called by Foxit Reader when the font name is changed.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

lpwsFontName	[In] The font name.
--------------	---------------------

lpwsScriptName	[In] The script name.
----------------	-----------------------

**Return**

void

**Head file reference**

fr\_barExpT.h: 898

**Group**[FR\\_FormatToolCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRFormatToolOnFontSizeChanged****Syntax**

```
typedef void (*FRFormatToolOnFontSizeChanged)(  
    FS\_LPVOID clientData,  
    FS\_FLOAT flFontSize  
);
```

**Description**

It is called by Foxit Reader when the font size is changed.

**Parameter**

clientData	[In] The user-supplied data.
flFontSize	[In] The font size.

**Return**

void

**Head file reference**

fr\_barExpT.h: 911

**Group**[FR\\_FormatToolCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRFormatToolOnTextColorChanged****Syntax**

```
typedef void (*FRFormatToolOnTextColorChanged)(  
    FS\_LPVOID clientData,  
    COLORREF textColor  
);
```

**Description**

It is called by Foxit Reader when the text color is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

textColor	[In] The text color.
-----------	----------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 924

**Group**[FR\\_FormatToolCallbacksRec](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FRFormatToolOnLineColorChanged****Syntax**

```
typedef void (*FRFormatToolOnLineColorChanged)(  
    FS\_LPVVOID clientData,  
    COLORREF lineColor,  
    FS\_FLOAT bTransparent  
);
```

**Description**

It is called by Foxit Reader when the line color is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

lineColor	[In] The text color.
-----------	----------------------

---

bTransparent	[In] Indicates whether the line is transparent.
--------------	---

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 938

**Group**[FR\\_FormatToolCallbacksRec](#)

**Since**[SDK LATEEST VERSION > 2.0](#)**FRFormatToolOnFillColorChanged****Syntax**

```
typedef void (*FRFormatToolOnFillColorChanged)(  
    FS_LPVVOID clientData,  
    COLORREF FillColor,  
    FS_FLOAT bTransparent  
)
```

**Description**

It is called by Foxit Reader when the filling color is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

FillColor	[In] The text color.
-----------	----------------------

---

bTransparent	[In] Indicates whether the line is transparent.
--------------	---

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 952

**Group**[FR\\_FormatToolCallbacksRec](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FRFormatToolOnBoldChanged****Syntax**

```
typedef void (*FRFormatToolOnBoldChanged)(  
    FS_LPVVOID clientData,  
    FS_FLOAT bBold,  
    FS_FLOAT bEnabled  
)
```

**Description**

It is called by Foxit Reader when the bold setting is changed.

**Parameter**

clientData	[In] The user-supplied data.
bBold	[In] Indicates whether uses the bold.
bEnabled	[In] Indicates whether the bold setting button is enabled or not.

**Return**

void

**Head file reference**

fr\_barExpT.h: 966

**Group**

[FR\\_FormatToolCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRFormatToolOnItalicChanged****Syntax**

```
typedef void (*FRFormatToolOnItalicChanged)(  
    FS\_LPVVOID clientData,  
    FS\_FLOAT bItalic,  
    FS\_FLOAT bEnabled  
)
```

**Description**

It is called by Foxit Reader when the italic setting is changed.

**Parameter**

clientData	[In] The user-supplied data.
bItalic	[In] Indicates whether uses the italic.
bEnabled	[In] Indicates whether the italic setting button is enabled or not.

**Return**

void

**Head file reference**

fr\_barExpT.h: 980

**Group**[FR\\_FormatToolCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRFormatToolOnAlignChanged****Syntax**

```
typedef void (*FRFormatToolOnAlignChanged)(  
    FS\_LPVVOID clientData,  
    FS\_DWORD dwAlign  
)
```

**Description**

It is called by Foxit Reader when the alignment setting is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

dwAlign	[In] The alignment type.
---------	--------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 993

**Group**[FR\\_FormatToolCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRFormatToolOnCharSpaceChanged****Syntax**

```
typedef void (*FRFormatToolOnCharSpaceChanged)(  
    FS\_LPVVOID clientData,  
    FS\_FLOAT flSpace  
)
```

**Description**

It is called by Foxit Reader when the character space setting is changed.

**Parameter**

clientData	[In] The user-supplied data.
flSpace	[In] The character space.

**Return**

void

**Head file reference**

fr\_barExpT.h: 1006

**Group**

[FR\\_FormatToolCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRFormatToolOnCharHorzScaleChanged

**Syntax**

```
typedef void (*FRFormatToolOnCharHorzScaleChanged)(  
    FS\_LPVOID clientData,  
    FS\_INT32 nScale  
);
```

**Description**

It is called by Foxit Reader when the character horizon scale setting is changed.

**Parameter**

clientData	[In] The user-supplied data.
nScale	[In] The character horizon scale.

**Return**

void

**Head file reference**

fr\_barExpT.h: 1019

**Group**

[FR\\_FormatToolCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRFormatToolOnLineLeadingChanged

**Syntax**

```
typedef void (*FRFormatToolOnLineLeadingChanged)(  
    FS\_LPVOID clientData,  
    FS\_FLOAT flLineLeading  
)
```

**Description**

It is called by Foxit Reader when the line leading setting is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

flLineLeading	[In] Indicates whether uses the leading line setting.
---------------	---

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1032

**Group**

[FR\\_FormatToolCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRFormatToolOnUnderlineChanged

**Syntax**

```
typedef void (*FRFormatToolOnUnderlineChanged)(  
    FS\_LPVOID clientData,  
    FS\_FLOAT bUnderline  
)
```

**Description**

It is called by Foxit Reader when the underline setting is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

bUnderline	[In] Indicates whether uses the underline setting.
------------	--

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1045

**Group**[FR\\_FormatToolCallbacksRec](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FRFormatToolOnCrossChanged****Syntax**

```
typedef void (*FRFormatToolOnCrossChanged)(
    FS_LPVVOID clientData,
    FS_FLOAT bCross
);
```

**Description**

It is called by Foxit Reader when the cross setting is changed.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

bCross	[In] Indicates whether uses the cross setting.
--------	--

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1058

**Group**[FR\\_FormatToolCallbacksRec](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FRFormatToolOnSuperScriptChanged****Syntax**

```
typedef void (*FRFormatToolOnSuperScriptChanged)(
```

```
FS_LPVVOID clientData,  
FS_FLOAT bSuperSet  
);
```

**Description**

It is called by Foxit Reader when the superscript setting is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
bSuperSet	[In] Indicates whether uses the superscript setting.

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1071

**Group**

[FR\\_FormatToolCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRFormatToolOnSubScriptChanged****Syntax**

```
typedef void (*FRFormatToolOnSubScriptChanged)(  
    FS_LPVVOID clientData,  
    FS_FLOAT bSubSet  
);
```

**Description**

It is called by Foxit Reader when the subscript setting is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
bSubSet	[In] Indicates whether uses the subscript setting.

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1084

**Group**

[FR\\_FormatToolCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolOnInDentChanged**

**Syntax**

```
typedef void (*FRFormatToolOnInDentChanged)(  
    FS\_LPVVOID clientData  
);
```

**Description**

It is called by Foxit Reader when the indent setting is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1096

**Group**

[FR\\_FormatToolCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolOnDeDentChanged**

**Syntax**

```
typedef void (*FRFormatToolOnDeDentChanged)(  
    FS\_LPVVOID clientData  
);
```

**Description**

It is called by Foxit Reader when the dedent setting is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1108

**Group**[FR\\_FormatToolCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRFormatToolOnWordSpaceChanged****Syntax**

```
typedef void (*FRFormatToolOnWordSpaceChanged)(
    FS_LPVVOID clientData,
    FS_FLOAT fWordSpace
);
```

**Description**

It is called by Foxit Reader when the word space setting is changed.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

fWordSpace	[In] Indicates whether uses the word space setting.
------------	---

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1121

**Group**[FR\\_FormatToolCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRFormatToolOnArrangeAlign****Syntax**

```
typedef void (*FRFormatToolOnArrangeAlign)(
```

```
FS_LPVOID clientData,  
FRFormatToolArrangeAlignInfo nAlign  
);
```

**Description**

It is called by Foxit Reader when the alignment of the arrange info is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
nAlign	[In] The alignment of the arrange info.

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1134

**Group**

[FR\\_FormatToolCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRFormatToolOnArrangeCenter****Syntax**

```
typedef void (*FRFormatToolOnArrangeCenter)(  
    FS_LPVOID clientData,  
    FRFormatToolArrangeCenterInfo nCenter  
);
```

**Description**

It is called by Foxit Reader when the page centering of the arrange info is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
nCenter	[In] The page centering of the arrange info.

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1147

**Group**

[FR\\_FormatToolCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

## FRFormatToolOnArrangeDistribute

**Syntax**

```
typedef void (*FRFormatToolOnArrangeDistribute)(  
    FS_LPVVOID clientData,  
    FRFormatToolArrangeDistributeInfo nDistribute  
>);
```

**Description**

It is called by Foxit Reader when the distribution of the arrange info is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

nDistribute	[In] The distribution of the arrange info.
-------------	--

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1160

**Group**

[FR\\_FormatToolCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

## FRFormatToolGetArrangeEnable

**Syntax**

```
typedef void (*FRFormatToolGetArrangeEnable)(  
    FS_LPVVOID clientData,  
    FRFormatToolArrangeInfo nInfo,  
    FS_BOOL* outEnable  
>);
```

**Description**

It is called by Foxit Reader to determine whether the arrange setting is enabled or not.

**Parameter**

clientData	[In] The user-supplied data.
nInfo	[In] The type of the arrange info.
outEnable	[Out] It receives TRUE if the arrange setting is enabled.

**Return**

void

**Head file reference**

fr\_barExpT.h: 1174

**Group**

[FR\\_FormatToolCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRFormatToolGetLineColorEnableNoColor****Syntax**

```
typedef void (*FRFormatToolGetLineColorEnableNoColor)(  
    FS_LPVVOID clientData,  
    FS_BOOL* outEnable  
>);
```

**Description**

It is called by Foxit Reader to determine whether the line color setting is enabled or not.

**Parameter**

clientData	[In] The user-supplied data.
outEnable	[Out] It receives TRUE if the line color setting is enabled.

**Return**

void

**Head file reference**

fr\_barExpT.h: 1187

**Group**

[FR\\_FormatToolCallbacksRec](#)**Since**[SDK LATEST VERSION > 2.0](#)**FRFormatToolGetFillColorEnableNoColor****Syntax**

```
typedef void (*FRFormatToolGetFillColorEnableNoColor)(
    FS\_LPVVOID clientData,
    FS\_BOOL* outEnable
);
```

**Description**

It is called by Foxit Reader to determine whether the filling color setting is enabled or not.

**Parameter**

clientData	[In] The user-supplied data.
outEnable	[Out] It receives TRUE if the filling color setting is enabled.

**Return**

void

**Head file reference**

fr\_barExpT.h: 1200

**Group**[FR\\_FormatToolCallbacksRec](#)**Since**[SDK LATEST VERSION > 2.0](#)**FRFormatToolOnWritingDirChanged****Syntax**

```
typedef void (*FRFormatToolOnWritingDirChanged)(
    FS\_LPVVOID clientData,
    FRFormatToolWritingDirection eDir
);
```

**Description**

It is called by Foxit Reader when the writing direction is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

eDir	[In] It the writing direction.
------	--------------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1213

**Group**[FR\\_FormatToolCallbacksRec](#)**Since**[SDK LATEEST VERSION > 2.1.0.4](#)**FRFormatToolOnOpacityChanged****Syntax**

```
typedef void (*FRFormatToolOnOpacityChanged)(  
    FS\_LPVVOID clientData,  
    FS\_INT32 nOpacity  
);
```

**Description**

It is called by Foxit Reader when the opacity setting is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

nOpacity	[In] The opacity value.
----------	-------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1226

**Group**[FR\\_FormatToolCallbacksRec](#)**Since**[SDK LATEEST VERSION > 2.1.0.4](#)**FRFormatToolOnOpacityScroll**

**Syntax**

```
typedef void (*FRFormatToolOnOpacityScroll)(  
    FS\_LPVOID clientData,  
    FS\_INT32 nOpacity  
) ;
```

**Description**

It is called by Foxit Reader when the opacity setting is scrolling.

**Parameter**

clientData	[In] The user-supplied data.
nOpacity	[In] The opacity value.

**Return**

void

**Head file reference**

fr\_barExpT.h: 1239

**Group**

[FR\\_FormatToolCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

## Functions

### Functions summary

**[FRFormatToolsActivateFormatCategory](#)**

Activates the format context category.

**[FRFormatToolsCleanOwnerFontNameArr](#)**

Cleans the owner font names.

**[FRFormatToolsCountFontList](#)**

Gets the count of the font.

**[FRFormatToolsEnable](#)**

Sets to enable the format tools or not.

**[FRFormatToolsEnableButton](#)**

Sets to enable the specified button or not.

**[FRFormatToolsFindFontName](#)**

Get the index of the specified font name.

**[FRFormatToolsGet](#)**

Gets the format tool object. The format tool is used to set the format of the PDF object and text. You can set the format like font name, font size, color and so on.

**[FRFormatToolsGetAlign](#)**

Gets the alignment of the format tools.

**FRFormatToolsGetCharHorzScale**

Gets the character horizon scale of the format tools.

**FRFormatToolsGetCharSpace**

Gets the character space of the format tools.

**FRFormatToolsGetCross**

Checks whether uses the cross setting or not.

**FRFormatToolsGetFillColor**

Gets fill color of the format tools.

**FRFormatToolsGetFontFaceName**

Gets the font face name of the format tools.

**FRFormatToolsGetFontListItem**

Gets the font name by index.

**FRFormatToolsGetFontName**

Gets the font name of the format tools.

**FRFormatToolsGetFontSize**

Gets the font size of the format tools.

**FRFormatToolsGetLineColor**

Gets the line color of the format tools.

**FRFormatToolsGetLineLeading**

Checks whether uses the leading line setting or not.

**FRFormatToolsGetSubScript**

Checks whether uses the subscript setting or not.

**FRFormatToolsGetSuperScript**

Checks whether uses the superscript setting or not.

**FRFormatToolsGetTextColor**

Gets the text color of the format tools.

**FRFormatToolsGetUnderline**

Checks whether uses the underline line setting or not.

**FRFormatToolsGetWordSpace**

Gets the word space of the format tools.

**FRFormatToolsGetWritingDirection**

Gets the writing direction on the format tools.

**FRFormatToolsHideButton**

Sets to hide the specified ID or not.

**FRFormatToolsIsBold**

Checks whether uses the bold setting or not.

**FRFormatToolsIsButtonEnabled**

Checks whether the specified button is enabled or not.

**FRFormatToolsIsButtonVisible**

Checks whether the specified button is visible or not.

**FRFormatToolsIsEnabled**

Checks whether the format tools is enabled or not.

**FRFormatToolsIsItalic**

Checks whether uses the italic setting or not.

**FRFormatToolsIsVisible**

Checks whether the format tools are visible or not.

**FRFormatToolsReleaseEvent**

Releases the event handle returned by [FRFormatToolsSetEvent](#) .

**FRFormatToolsSetAlign**

Sets the alignment of the format tools.

**FRFormatToolsSetBold**

Sets whether uses the bold and whether the bold setting button is enabled.

**FRFormatToolsSetCharHorzScale**

Sets the character horizon scale of the format tools.

**FRFormatToolsSetCharSpace**

Sets the character space of the format tools.

**FRFormatToolsSetCross**

Sets whether uses the cross setting.

**FRFormatToolsSetEvent**

Sets the callback functions that are called by Foxit Reader when the events occur.

**FRFormatToolsSetFillColor**

Sets the fill color of the format tools.

**FRFormatToolsSetFontName**

Sets the font name of the format tools.

**FRFormatToolsSetFontNameFirst**

Sets whether you set the font name the first time or not.

**FRFormatToolsSetFontSize**

Sets the font size of the format tools.

**FRFormatToolsSetFontSizeFirst**

Sets whether you set the font size the first time or not.

**FRFormatToolsSetFormatContextTitle**

Sets the title of the format context category. This interface is valid in ribbon mode.

**FRFormatToolsSetItalic**

Sets whether uses the italic and whether the italic setting button is enabled.

**FRFormatToolsSetLineColor**

Sets the line color of the format tools.

**FRFormatToolsSetLineLeading**

Sets whether uses the leading line setting.

**FRFormatToolsSetOwnerFontNameArr**

Sets the owner font names except for the system font name.

**FRFormatToolsSetSubScript**

Sets whether uses the subscript setting.

**FRFormatToolsSetSuperScript**

Sets whether uses the superscript setting.

**FRFormatToolsSetTextColor**

Sets the text color of the format tools.

**FRFormatToolsSetUnderline**

Sets whether uses the underline setting.

**FRFormatToolsSetWordSpace**

Sets the word space of the format tools.

**FRFormatToolsSetWritingDirection**

Sets the writing direction on the format tools.

**FRFormatToolsShow**

Sets to show or hide the format tools.

## Functions detail

### **FRFormatToolsActivateFormatCategory**

**Syntax**

```
void FRFormatToolsActivateFormatCategory (
```



```
    FR\_FormatTools formatTools,  
    FRFormatToolContextCategoryType eCateType  
);
```

**Description**

Activates the format context category.

**Parameter**

---

formatTools	[In] The input format tools object.
eCateType	[In] Sets the type of format to be operated.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5669

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsCleanOwnerFontNameArr****Syntax**

```
void FRFormatToolsCleanOwnerFontNameArr (  
    FR\_FormatTools formatTools,  
    FS\_WideStringArray array  
)
```

**Description**

Cleans the owner font names.

**Parameter**

---

formatTools	[In] The input format tools object.
array	[In] The input font name array.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5608



**Related method**[FRFormatToolsSetOwnerFontNameArr](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FRFormatToolsCountFontList****Syntax**

```
FS_INT32 FRFormatToolsCountFontList (
    FR\_FormatTools formatTools
);
```

**Description**

Gets the count of the font.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

**Return**

The count of the font.

**Head file reference**

fr\_barTempl.h: 5632

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRFormatToolsEnable****Syntax**

```
void FRFormatToolsEnable (
    FR\_FormatTools formatTools,
    FS\_BOOL bEnable
);
```

**Description**

Sets to enable the format tools or not.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---



---

bEnable	[In] Sets TRUE to enable the format tools, otherwise not.
---------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5534

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRFormatToolsEnableButton****Syntax**

```
void FRFormatToolsEnableButton (
    FR\_FormatTools formatTools,
    FS\_INT32 nID,
    FS\_BOOL bEnabled
);
```

**Description**

Sets to enable the specified button or not.

**Parameter**


---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

nID	[In] The specified ID.
-----	------------------------

---

bEnabled	[In] Sets TRUE to enable the button, otherwise not.
----------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5583

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRFormatToolsFindFontName**

**Syntax**

```
FS_INT32 FRFormatToolsFindFontName (
    FR\_FormatTools formatTools,
    FS\_LPCWSTR lpwsFontName
);
```

**Description**

Get the index of the specified font name.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

lpwsFontName	[In] The specified font name.
--------------	-------------------------------

---

**Return**

The index of the font name.

**Head file reference**

fr\_barTempl.h: 5620

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsGet****Syntax**

```
FR_FormatTools FRFormatToolsGet (void );
```

**Description**

Gets the format tool object. The format tool is used to set the format of the PDF object and text. You can set the format like font name, font size, color and so on.

**Return**

The format tool object.

**Head file reference**

fr\_barTempl.h: 5076

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsGetAlign**

**Syntax**

```
FS_DWORD FRFormatToolsGetAlign (
    FR\_FormatTools formatTools
);
```

**Description**

Gets the alignment of the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

**Return**

The alignment of the format tools.

**Head file reference**

fr\_barTempl.h: 5378

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsGetCharHorzScale****Syntax**

```
FS_INT32 FRFormatToolsGetCharHorzScale (
    FR\_FormatTools formatTools
);
```

**Description**

Gets the character horizon scale of the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

**Return**

The character horizon scale of the format tools.

**Head file reference**

fr\_barTempl.h: 5400

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRFormatToolsGetCharSpace

### Syntax

```
FS_FLOAT FRFormatToolsGetCharSpace (
    FR\_FormatTools formatTools
);
```

### Description

Gets the character space of the format tools.

### Parameter

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

### Return

The character space of the format tools.

### Head file reference

fr\_barTempl.h: 5389

### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRFormatToolsGetCross

### Syntax

```
FS_BOOL FRFormatToolsGetCross (
    FR\_FormatTools formatTools
);
```

### Description

Checks whether uses the cross setting or not.

### Parameter

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

### Return

[TRUE](#) if uses the cross setting, otherwise not.

### Head file reference

fr\_barTempl.h: 5433

### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)



## FRFormatToolsGetFillColor

### Syntax

```
FS_BOOL FRFormatToolsGetFillColor (
    FR\_FormatTools formatTools,
    COLORREF* outFillColor
);
```

### Description

Gets fill color of the format tools.

### Parameter

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

outFillColor	[Out] It receives the fill color.
--------------	-----------------------------------

---

### Return

[TRUE](#) for success, otherwise not.

### Head file reference

fr\_barTempl.h: 5366

### Related method

#### Since

[SDK LATEEST VERSION > 2.0](#)

## FRFormatToolsGetFontFaceName

### Syntax

```
void FRFormatToolsGetFontFaceName (
    FR\_FormatTools formatTools,
    FS\_LPCWSTR lpwsScriptName,
    FS\_WideString* outFontFaceName
);
```

### Description

Gets the font face name of the format tools.

### Parameter

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

lpwsScriptName	[In] The input font script name.
----------------	----------------------------------

---

---

outFontFaceName	[Out] It receives the font face name.
-----------------	---------------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5705

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRFormatToolsGetFontListItem****Syntax**

```
void FRFormatToolsGetFontListItem (
    FR\_FormatTools formatTools,
    FS\_INT32 nIndex,
    FS\_WideString* outItemName
);
```

**Description**

Gets the font name by index.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

---

nIndex	[In] The specified index.
--------	---------------------------

---

---

outItemName	[Out] It receives the font name.
-------------	----------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5643

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsGetFontName**

**Syntax**

```
void FRFormatToolsGetFontName (
    FR\_FormatTools formatTools,
    FS\_WideString* outFontName
);
```

**Description**

Gets the font name of the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

outFontName	[Out] It receives the font name.
-------------	----------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5319

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsGetFontSize****Syntax**

```
FS_FLOAT FRFormatToolsGetFontSize (
    FR\_FormatTools formatTools
);
```

**Description**

Gets the font size of the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

**Return**

The font size.

**Head file reference**

fr\_barTempl.h: 5331

**Related method**

**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRFormatToolsGetLineColor

**Syntax**

```
FS_BOOL FRFormatToolsGetLineColor (
    FR\_FormatTools formatTools,
    COLORREF* outLineColor
);
```

**Description**

Gets the line color of the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

outLineColor	[Out] It receives the line color.
--------------	-----------------------------------

---

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

fr\_barTempl.h: 5354

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRFormatToolsGetLineLeading

**Syntax**

```
FS_FLOAT FRFormatToolsGetLineLeading (
    FR\_FormatTools formatTools
);
```

**Description**

Checks whether uses the leading line setting or not.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---



**Return**

[TRUE](#) if uses the leading line setting, otherwise not.

**Head file reference**

fr\_barTempl.h: 5411

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRFormatToolsGetSubScript****Syntax**

```
FS_BOOL FRFormatToolsGetSubScript (
    FR\_FormatTools formatTools
);
```

**Description**

Checks whether uses the subscript setting or not.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

**Return**

[TRUE](#) if uses the subscript setting, otherwise not.

**Head file reference**

fr\_barTempl.h: 5455

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRFormatToolsGetSuperScript****Syntax**

```
FS_BOOL FRFormatToolsGetSuperScript (
    FR\_FormatTools formatTools
);
```

**Description**

Checks whether uses the superscript setting or not.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

**Return**

[TRUE](#) if uses the superscript setting, otherwise not.

**Head file reference**

fr\_barTempl.h: 5444

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsGetTextColor****Syntax**

```
FS_BOOL FRFormatToolsGetTextColor (
    FR\_FormatTools formatTools,
    COLORREF* outTextColor
);
```

**Description**

Gets the text color of the format tools.

**Parameter**


---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

outTextColor	[Out] It receives the text color.
--------------	-----------------------------------

---

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

fr\_barTempl.h: 5342

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsGetUnderline****Syntax**

```
FS_BOOL FRFormatToolsGetUnderline (
    FR\_FormatTools formatTools
);
```

**Description**

Checks whether uses the underline line setting or not.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

**Return**

[TRUE](#) if uses the underline setting, otherwise not.

**Head file reference**

fr\_barTempl.h: 5422

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRFormatToolsGetWordSpace****Syntax**

```
FS_FLOAT FRFormatToolsGetWordSpace (
    FR FormatTools formatTools
);
```

**Description**

Gets the word space of the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

**Return**

The word space.

**Head file reference**

fr\_barTempl.h: 5488

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRFormatToolsGetWritingDirection**

**Syntax**

```
FS_BOOL FRFormatToolsGetWritingDirection (
    FR\_FormatTools formatTools,
    FRFormatToolWritingDirection* outDir
);
```

**Description**

Gets the writing direction on the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

outDir	[Out] It receives the writing direction.
--------	--

---

**Return**

True for success, otherwise for failure.

**Head file reference**

fr\_barTempl.h: 5693

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRFormatToolsHideButton****Syntax**

```
void FRFormatToolsHideButton (
    FR\_FormatTools formatTools,
    FS\_INT32 nID,
    FS\_BOOL bHide
);
```

**Description**

Sets to hide the specified ID or not.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

nID	[In] The specified ID.
-----	------------------------

---

bHide	[In] Sets TRUE to hide the specified ID, otherwise not.
-------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5558

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRFormatToolsIsBold****Syntax**

```
FS_BOOL FRFormatToolsIsBold (
    FR\_FormatTools formatTools
);
```

**Description**

Checks whether uses the bold setting or not.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

**Return**

[TRUE](#) if uses the bold setting, otherwise not.

**Head file reference**

fr\_barTempl.h: 5466

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRFormatToolsIsButtonEnabled****Syntax**

```
FS_BOOL FRFormatToolsIsButtonEnabled (
    FR\_FormatTools formatTools,
    FS\_INT32 nID
);
```

**Description**

Checks whether the specified button is enabled or not.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

nID	[In] The specified ID.
-----	------------------------

---

**Return**

[TRUE](#) if the button is enabled, otherwise not.

**Head file reference**

fr\_barTempl.h: 5571

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRFormatToolsIsButtonVisible****Syntax**

```
FS_BOOL FRFormatToolsIsButtonVisible (
    FR\_FormatTools formatTools,
    FS\_INT32 nID
);
```

**Description**

Checks whether the specified button is visible or not.

**Parameter**


---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

nID	[In] The specified ID.
-----	------------------------

---

**Return**

[TRUE](#) if the button is visible, otherwise not.

**Head file reference**

fr\_barTempl.h: 5546

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRFormatToolsIsEnabled****Syntax**

```
FS_BOOL FRFormatToolsIsEnabled (
    FR\_FormatTools formatTools
);
```

**Description**

Checks whether the format tools is enabled or not.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

**Return**

[TRUE](#) if the format tools is enabled, otherwise not.

**Head file reference**

fr\_barTempl.h: 5523

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRFormatToolsIsItalic****Syntax**

```
FS_BOOL FRFormatToolsIsItalic (
    FR\_FormatTools formatTools
);
```

**Description**

Checks whether uses the italic setting or not.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

**Return**

[TRUE](#) if uses the italic setting, otherwise not.

**Head file reference**

fr\_barTempl.h: 5477

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

## FRFormatToolsIsVisible

### Syntax

```
FS_BOOL FRFormatToolsIsVisible (
    FR\_FormatTools formatTools
);
```

### Description

Checks whether the format tools are visible or not.

### Parameter

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

### Return

[TRUE](#) if the format tools are visible, otherwise not.

### Head file reference

fr\_barTempl.h: 5499

### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRFormatToolsReleaseEvent

### Syntax

```
void FRFormatToolsReleaseEvent (
    void* eventHandle
);
```

### Description

Releases the event handle returned by [FRFormatToolsSetEvent](#).

### Parameter

---

eventHandle	[In] The input event handle returned by <a href="#">FRFormatToolsSetEvent</a> .
-------------	---

---

### Return

void.

### Head file reference

fr\_barTempl.h: 5091

### Related method

[FRFormatToolsSetEvent](#)

**Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)

## FRFormatToolsSetAlign

**Syntax**

```
void FRFormatToolsSetAlign (
    FR\_FormatTools formatTools,
    FS\_DWORD dwAlign
);
```

**Description**

Sets the alignment of the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

dwAlign	[In] The input alignment.
---------	---------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5211

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRFormatToolsSetBold

**Syntax**

```
void FRFormatToolsSetBold (
    FR\_FormatTools formatTools,
    FS\_BOOL bBold,
    FS\_BOOL bEnabled
);
```

**Description**

Sets whether uses the bold and whether the bold setting button is enabled.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

---

bBold	[In] Indicates whether uses the bold.
bEnabled	[In] Indicates whether the bold setting button is enabled or not.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5185

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRFormatToolsSetCharHorzScale****Syntax**

```
void FRFormatToolsSetCharHorzScale (
    FR\_FormatTools formatTools,
    FS\_INT32 nScale
);
```

**Description**

Sets the character horizon scale of the format tools.

**Parameter**


---

formatTools	[In] The input format tools object.
nScale	[In] The input character horizon scale.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5235

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRFormatToolsSetCharSpace****Syntax**

```
void FRFormatToolsSetCharSpace (
    FR\_FormatTools formatTools,
    FS\_FLOAT flSpace
);
```

**Description**

Sets the character space of the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

flSpace	[In] The input character space.
---------	---------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5223

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsSetCross****Syntax**

```
void FRFormatToolsSetCross (
    FR\_FormatTools formatTools,
    FS\_BOOL bCross
);
```

**Description**

Sets whether uses the cross setting.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

bCross	[In] Indicates whether uses the cross setting.
--------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5295

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

## FRFormatToolsSetEvent

#### Syntax

```
void* FRFormatToolsSetEvent (
    FR\_FormatTools formatTools,
    FR\_FormatToolCallbacks callbacks
);
```

#### Description

Sets the callback functions that are called by Foxit Reader when the events occur.

#### Parameter

formatTools	[In] The input format tools object.
callbacks	[In] The callback functions. They are called by Foxit Reader when the events occur.

#### Return

The returned event handle. Releases it by [FRFormatToolsReleaseEvent](#).

#### Head file reference

fr\_barTempl.h: 5087

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

## FRFormatToolsSetFillColor

#### Syntax

```
void FRFormatToolsSetFillColor (
    FR\_FormatTools formatTools,
    COLORREF fillColor,
    FS\_BOOL bTransparent
);
```

#### Description

Sets the fill color of the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

fillColor	[In] The input fill color.
-----------	----------------------------

---

bTransparent	[In] Indicates whether the line is transparent.
--------------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5172

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRFormatToolsSetFontName****Syntax**

```
void FRFormatToolsSetFontName (
    FR\_FormatTools formatTools,
    FS\_LPCWSTR lpwsFontName
);
```

**Description**

Sets the font name of the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

lpwsFontName	[In] The input font name.
--------------	---------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5099

**Related method****Since**

## [SDK\\_LATEEST\\_VERSION > 2.0](#)

### FRFormatToolsSetFontNameFirst

#### Syntax

```
void FRFormatToolsSetFontNameFirst (
    FR\_FormatTools formatTools,
    FS\_BOOL bFirst
);
```

#### Description

Sets whether you set the font name the first time or not.

#### Parameter

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

bFirst	[In] Indicates whether you set the font name the first time.
--------	--

---

#### Return

void

#### Head file reference

fr\_barTempl.h: 5111

#### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

### FRFormatToolsSetFontSize

#### Syntax

```
void FRFormatToolsSetFontSize (
    FR\_FormatTools formatTools,
    FS\_FLOAT flFontSize
);
```

#### Description

Sets the font size of the format tools.

#### Parameter

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---



---

flFontSize	[In] The input font size.
------------	---------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5123

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRFormatToolsSetFontSizeFirst****Syntax**

```
void FRFormatToolsSetFontSizeFirst (
    FR\_FormatTools formatTools,
    FS\_BOOL bFirst
);
```

**Description**

Sets whether you set the font size the first time or not.

**Parameter**


---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

bFirst	[In] Indicates whether you set the font size the first time.
--------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5135

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRFormatToolsSetFormatContextTitle****Syntax**

```
void FRFormatToolsSetFormatContextTitle (
    FR\_FormatTools formatTools,
    FS\_LPCWSTR lpwsContextTitle,
    FRFormatToolContextCategoryType eCatType
```

---

);

**Description**

Sets the title of the format context category. This interface is valid in ribbon mode.

**Parameter**


---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

lpwsContextTitle	[In] The input title of the format context category.
------------------	--

---

eCateType	[In] Sets the type of format to be operated.
-----------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5656

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsSetItalic****Syntax**

```
void FRFormatToolsSetItalic (
    FR\_FormatTools formatTools,
    FS\_BOOL bItalic,
    FS\_BOOL bEnabled
);
```

**Description**

Sets whether uses the italic and whether the italic setting button is enabled.

**Parameter**


---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

bItalic	[In] Indicates whether uses the italic.
---------	---

---

bEnabled	[In] Indicates whether the italic setting button is enabled or not.
----------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5198

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsSetLineColor**

**Syntax**

```
void FRFormatToolsSetLineColor (
    FR\_FormatTools formatTools,
    COLORREF lineColor,
    FS\_BOOL bTransparent
);
```

**Description**

Sets the line color of the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

lineColor	[In] The input line color.
-----------	----------------------------

---

bTransparent	[In] Indicates whether the line is transparent.
--------------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5159

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsSetLineLeading**

**Syntax**

```
void FRFormatToolsSetLineLeading (
    FR\_FormatTools formatTools,
    FS\_FLOAT fLineLeading
```

);

**Description**

Sets whether uses the leading line setting.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

flLineLeading	[In] Indicates whether uses the leading line setting.
---------------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5247

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRFormatToolsSetOwnerFontNameArr****Syntax**

```
void FRFormatToolsSetOwnerFontNameArr (
    FR\_FormatTools formatTools,
    FS\_WideStringArray array
);
```

**Description**

Sets the owner font names except for the system font name.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

array	[In] The input font name array.
-------	---------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5596

**Related method**

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRFormatToolsSetSubScript

**Syntax**

```
void FRFormatToolsSetSubScript (
    FR\_FormatTools formatTools,
    FS\_BOOL bSubScript
);
```

**Description**

Sets whether uses the subscript setting.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

bSubScript	[In] Indicates whether uses the subscript setting.
------------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5271

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRFormatToolsSetSuperScript

**Syntax**

```
void FRFormatToolsSetSuperScript (
    FR\_FormatTools formatTools,
    FS\_BOOL bSuperScript
);
```

**Description**

Sets whether uses the superscript setting.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

---

bSuperScript	[In] Indicates whether uses the superscript setting.
--------------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5259

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsSetTextColor****Syntax**

```
void FRFormatToolsSetTextColor (
    FR\_FormatTools formatTools,
    COLORREF textColor
);
```

**Description**

Sets the text color of the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

---

textColor	[In] The input color value of the text.
-----------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5147

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsSetUnderline****Syntax**

```
void FRFormatToolsSetUnderline (
    FR\_FormatTools formatTools,
    FS\_BOOL bUnderline
);
```

**Description**

Sets whether uses the underline setting.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

bUnderline	[In] Indicates whether uses the underline setting.
------------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5283

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRFormatToolsSetWordSpace****Syntax**

```
void FRFormatToolsSetWordSpace (
    FR\_FormatTools formatTools,
    FS\_FLOAT fWordSpace
);
```

**Description**

Sets the word space of the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

fWordSpace	[In] Indicates whether uses the word space setting.
------------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5307

**Related method**

**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRFormatToolsSetWritingDirection

**Syntax**

```
void FRFormatToolsSetWritingDirection (
    FR\_FormatTools formatTools,
    FRFormatToolWritingDirection eDir
);
```

**Description**

Sets the writing direction on the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

eDir	[In] The input writing direction.
------	-----------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5681

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

## FRFormatToolsShow

**Syntax**

```
void FRFormatToolsShow (
    FR\_FormatTools formatTools,
    FS\_BOOL bShow,
    FRFormatToolContextCategoryType eCateType
);
```

**Description**

Sets to show or hide the format tools.

**Parameter**

---

formatTools	[In] The input format tools object.
-------------	-------------------------------------

---

---

bShow	[In] Sets TRUE to show the format tools and sets FALSE to hide it.
-------	--

---

eCateType	[In] Sets the type of format to be operated.
-----------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5510

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)

## FR\_FuncBtn

[\*\*Return from Used by\*\*](#)

### Description

The plug-in can add a button to the left navigation panel bar.

### Returned from

[\*\*FRFuncBtnCreate\*\*](#)

### Used by

[\*\*FRFuncBtnAddToPanel\*\*](#)  
[\*\*FRFuncBtnAddToTabBand\*\*](#)  
[\*\*FRFuncBtnGetClientData\*\*](#)  
[\*\*FRFuncBtnGetName\*\*](#)  
[\*\*FRFuncBtnGetRect\*\*](#)  
[\*\*FRFuncBtnGetToolTip\*\*](#)  
[\*\*FRFuncBtnSetClientData\*\*](#)  
[\*\*FRFuncBtnSetComputeEnabledProc\*\*](#)  
[\*\*FRFuncBtnSetComputeMarkedProc\*\*](#)  
[\*\*FRFuncBtnSetExecuteProc\*\*](#)  
[\*\*FRFuncBtnSetName\*\*](#)  
[\*\*FRFuncBtnSetToolTip\*\*](#)  
[\*\*FRFuncBtnUpdateImage\*\*](#)

## Functions

### Functions summary

[\*\*FRFuncBtnAddToPanel\*\*](#)

Adds the function button to the left navigation panel bar.

**FRFuncBtnAddToTabBand**

Adds the function button to the tab band.

**FRFuncBtnCreate**

Creates a function button object. Then invoke [FRFuncBtnAddToPanel](#) to show the function button.

**FRFuncBtnGetClientData**

Gets the pointer to the client data.

**FRFuncBtnGetName**

Gets the name of the function button.

**FRFuncBtnGetRect**

Gets the rectangle of the function button.

**FRFuncBtnGetToolTip**

Gets the tooltip of the function button.

**FRFuncBtnSetClientData**

Sets the client data.

**FRFuncBtnSetComputeEnabledProc**

Sets a [FRComputeEnabledProc](#) () associated with the function button. This routine determines whether a function button can be selected.

**FRFuncBtnSetComputeMarkedProc**

Sets a [FRComputeMarkedProc](#) () associated with the function button. This routine determines whether a function button can be marked.

**FRFuncBtnSetExecuteProc**

Sets the callback that will be invoked when the button is clicked.

**FRFuncBtnSetName**

Sets the name of the function button.

**FRFuncBtnSetToolTip**

Sets the tooltip of the function button.

**FRFuncBtnUpdateImage**

Updates the icon of the function button.

## Functions detail

### FRFuncBtnAddToPanel

#### Syntax

```
void FRFuncBtnAddToPanel (
    FR\_FuncBtn pFuncBtn,
    FS\_BOOL bShowBottom
);
```

#### Description

Adds the function button to the left navigation panel bar.

#### Parameter

pFuncBtn	[In] The input function button object.
----------	--

bShowBottom	[In] Shows the function button to the bottom or not.
-------------	--

#### Return

void.

**Head file reference**

fr\_barTempl.h: 5851

**Related method**

[FRFuncBtnCreate](#)

**Since**

[SDK LATEEST VERSION > 7.1](#)

**FRFuncBtnAddToTabBand****Syntax**

```
void FRFuncBtnAddToTabBand (
    FR\_FuncBtn pFuncBtn,
    FS\_BOOL bLeft
);
```

**Description**

Adds the function button to the tab band.

**Parameter**

---

pFuncBtn	[In] The input function button object.
----------	--

---

bLeft	[In] Shows the function button to the left or not.
-------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6012

**Related method**

[FRFuncBtnCreate](#)

**Since**

[SDK LATEEST VERSION > 7.2.2](#)

**FRFuncBtnCreate****Syntax**

```
FR_FuncBtn FRFuncBtnCreate (
    void* pParentWnd,
    FS\_LPCSTR lpsName,
    FS\_LPCWSTR lpwsToolTip,
    FS\_DIBitmap pBitmap
```

);

**Description**

Creates a function button object. Then invoke [FRFuncBtnAddToPanel](#) to show the function button.

**Parameter**

pParentWnd	[In] The parent window passed from <a href="#">PILoadStatusBarUI</a> .
lpsName	[In] Specifies the name of the function button.
lpwsToolTip	[In] Specifies the tooltip of the function button.
pBitmap	[In] Specifies the icon of the function button.

**Return**

A function button object.

**Head file reference**

fr\_barTempl.h: 5850

**Related method**

[FRFuncBtnAddToPanel](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRFuncBtnGetClientData****Syntax**

```
void* FRFuncBtnGetClientData (
    FR\_FuncBtn pFuncBtn
);
```

**Description**

Gets the pointer to the client data.

**Parameter**

pFuncBtn	[In] The input function button object.
----------	--

**Return**

Pointer to the client data.

**Head file reference**

fr\_barTempl.h: 5876

**Related method****Since**[SDK LATEEST VERSION > 7.1](#)**FRFuncBtnGetName****Syntax**

```
void FRFuncBtnGetName (
    FR\_FuncBtn pFuncBtn,
    FS\_ByteString* outName
);
```

**Description**

Gets the name of the function button.

**Parameter**

---

pFuncBtn	[In] The input function button object.
----------	--

---

outName	[Out] It receives the name of the function button.
---------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 5936

**Related method****Since**[SDK LATEEST VERSION > 7.1](#)**FRFuncBtnGetRect****Syntax**

```
void FRFuncBtnGetRect (
    FR\_FuncBtn pFuncBtn,
    FS\_Rect* outRt
);
```

**Description**

Gets the rectangle of the function button.

**Parameter**


---

pFuncBtn	[In] The input function button object.
----------	--

---

outRt	[Out] It receives the rectangle of the function button.
-------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6000

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRFuncBtnGetToolTip****Syntax**

```
void FRFuncBtnGetToolTip (
    FR\_FuncBtn pFuncBtn,
    FS\_WideString* outTooltip
);
```

**Description**

Gets the tooltip of the function button.

**Parameter**


---

pFuncBtn	[In] The input function button object.
----------	--

---

outTooltip	[Out] It receives the tooltip of the function button.
------------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 5900

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRFuncBtnSetClientData**

**Syntax**

```
void FRFuncBtnSetClientData (
    FR\_FuncBtn pFuncBtn,
    void* pData,
    FRFreeDataProc callback
);
```

**Description**

Sets the client data.

**Parameter**

---

pFuncBtn	[In] The input function button object.
----------	--

---

pData	[In] The input client data.
-------	-----------------------------

---

callback	[In] The callback will be invoked to free the client data.
----------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 5887

**Related method****Since**

[SDK LATEEST VERSION > 7.1](#)

**FRFuncBtnSetComputeEnabledProc****Syntax**

```
void FRFuncBtnSetComputeEnabledProc (
    FR\_FuncBtn pFuncBtn,
    FRComputeEnabledProc proc
);
```

**Description**

Sets a [FRComputeEnabledProc](#) () associated with the function button. This routine determines whether a function button can be selected.

**Parameter**

---

pFuncBtn	[In] The input function button object.
----------	--

---

---

proc	[In] A user-supplied procedure to call whenever Reader needs to know whether a button should be enabled.
------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 5960

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRFuncBtnSetComputeMarkedProc****Syntax**

```
void FRFuncBtnSetComputeMarkedProc (
    FR\_FuncBtn pFuncBtn,
    FRComputeMarkedProc proc
);
```

**Description**

Sets a [FRComputeMarkedProc](#) () associated with the function button. This routine determines whether a function button can be marked.

**Parameter**


---

pFuncBtn	[In] The input function button object.
----------	--

---



---

proc	[In] A user-supplied procedure to call whenever Reader needs to know whether a button should be marked.
------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 5974

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRFuncBtnSetExecuteProc****Syntax**

```
void FRFuncBtnSetExecuteProc (
```



```
FR_FuncBtn pFuncBtn,  
FRExecuteProc proc  
);
```

**Description**

Sets the callback that will be invoked when the button is clicked.

**Parameter**

---

pFuncBtn	[In] The input function button object.
proc	[In] It callback will be invoked when the button is clicked.

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 5948

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRFuncBtnSetName****Syntax**

```
void FRFuncBtnSetName (  
    FR_FuncBtn pFuncBtn,  
    FS_LPCSTR lpsButtonName  
) ;
```

**Description**

Sets the name of the function button.

**Parameter**

---

pFuncBtn	[In] The input function button object.
lpsButtonName	[In] The input name of the function button.

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 5924

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRFuncBtnSetToolTip****Syntax**

```
void FRFuncBtnSetToolTip (
    FR\_FuncBtn pFuncBtn,
    FS\_LPCWSTR lpwsToolTip
);
```

**Description**

Sets the tooltip of the function button.

**Parameter**

---

<a href="#">pFuncBtn</a>	[In] The input function button object.
--------------------------	--

---

<a href="#">lpwsToolTip</a>	[In] The input tooltip of the function button.
-----------------------------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 5912

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRFuncBtnUpdateImage****Syntax**

```
void FRFuncBtnUpdateImage (
    FR\_FuncBtn pFuncBtn,
    FS\_DIBitmap pBitmap
);
```

**Description**

Updates the icon of the function button.

**Parameter**

---

pFuncBtn	[In] The input function button object.
----------	--

---

pBitmap	[In] The input bitmap you want to update.
---------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 5988

**Related method****Since**[SDK LATEEST VERSION > 7.1](#)

## FR\_HtmlMgr

**[Return from Used by](#)****Description**

The [FR\\_HtmlMgr](#) object is used to manage the HTML windows in *Foxit Reader* . See [FRHTMLMgrGet](#) .

**Returned from****[FRHTMLMgrGet](#)****Used by**

[FRHTMLMgrGetActiveHtmlView](#)  
[FRHTMLMgrGetHtmlViewFromHandle](#)  
[FRHTMLMgrOpenHTMLDocument](#)  
[FRHTMLMgrOpenHTMLDocument2](#)  
[FRHTMLMgrOpenHTMLFromNewTab](#)  
[FRHTMLMgrRegisterFoxitBrowserEventHandler](#)  
[FRHTMLMgrRegisterHTMLEventHandler](#)  
[FRHTMLMgrSetFavoritesLink](#)  
[FRHTMLMgrSetFoxitBrowserHome](#)

**Callbacks****Callbacks summary****[FRFoxitBrowserEventRelease](#)**

A callback for Foxit browser event handler.

**[FRFoxitBrowserEventOnDelFavLinks](#)**

A callback for Foxit browser event handler.

**[FRHTMLEventRelease](#)**

A callback for HTML window event handler.

**[FRHTMLEventOnDocFrameCreate](#)**

A callback for HTML window event handler.

**[FRHTMLEventOnDocViewBeforeNavigate2](#)**

A callback for HTML window event handler.

**[FRHTMLEventOnDocViewBeforeDestory](#)**

A callback for HTML window event handler.

**[FRHTMLEventOnDocViewNewWindow3](#)**

A callback for HTML window event handler.

**[FRHTMLEventOnWillFrameClose](#)**

A callback for HTML window event handler.

**[FRHTMLEventOnDispatchFun](#)**

A callback for HTML window event handler.

## Callbacks detail

### **[FRFoxitBrowserEventRelease](#)**

#### **Syntax**

```
typedef void (*FRFoxitBrowserEventRelease)(  
    FS\_LPVVOID clientData  
);
```

#### **Description**

A callback for Foxit browser event handler. It is called when the event handler is released.

#### **Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

#### **Return**

#### **Head file reference**

[fr\\_appExpT.h: 6329](#)

#### **Group**

**[FR\\_FoxitBrowserEventCallbacksRec](#)**

#### **Since**

[SDK LATEEST VERSION > 7.2](#)

### **[FRFoxitBrowserEventOnDelFavLinks](#)**

#### **Syntax**

```
typedef void (*FRFoxitBrowserEventOnDelFavLinks)(  
    FS\_LPVVOID clientData,  
    FS\_LPCWSTR lpwsUrl  
);
```

**Description**

A callback for Foxit browser event handler. It is called when a favorite link is deleted.

**Parameter**

---

clientData	[In] The user-supplied data.
lpwsUrl	[In] The favorite link that is deleted.

---

**Return****Head file reference**

fr\_appExpT.h: 6343

**Group**

[FR\\_FoxitBrowserEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2](#)

**FRHTMLEventRelease****Syntax**

```
typedef void (*FRHTMLEventRelease)(  
    FS_LPVVOID clientData  
)
```

**Description**

A callback for HTML window event handler. It is called when the event handler is released.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 6197

**Group**

[FR\\_HTMLEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRHTMLEventOnDocFrameCreate**

**Syntax**

```
typedef void (*FRHTMLEventOnDocFrameCreate)(  
    FS_LPVVOID clientData,  
    HWND hFrameWnd,  
    HWND hView,  
    FS_BOOL bNeedCreatePanel  
)
```

**Description**

A callback for HTML window event handler. It is called when the HTML doc frame is created.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

hFrameWnd	[In] The HTML child frame window created.
-----------	---

hView	[In] The HTML view created.
-------	-----------------------------

bNeedCreatePanel	[In] Whether needs to create the navigation panel.
------------------	--

**Return****Head file reference**

fr\_appExpT.h: 6213

**Group**

[FR HTMLEventCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 7.1](#)

**FRHTMLEventOnDocViewBeforeNavigate2****Syntax**

```
typedef FS_BOOL (*FRHTMLEventOnDocViewBeforeNavigate2)(  
    FS_LPVVOID clientData,  
    HWND hView,  
    FS_LPCWSTR lpURL  
)
```

**Description**

A callback for HTML window event handler. It is called by the framework to cause an event to fire before a navigation occurs in the web browser.

**Parameter**

---

clientData	[In] The user-supplied data.
hView	[In] The HTML view.
lpURL	[In] Pointer to a string containing the URL to navigate to.

---

**Return**

TRUE if the plug-in processes the event and the framework will not dispatch the event.

**Head file reference**

fr\_appExpT.h: 6229

**Group**

[FR\\_HTMLEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRHTMLEventOnDocViewBeforeDestory****Syntax**

```
typedef void (*FRHTMLEventOnDocViewBeforeDestory)(  
    FS\_LVOID clientData,  
    HWND hView,  
    FS\_LPCWSTR lpURL  
>;
```

**Description**

A callback for HTML window event handler. It is called when the HTML doc view is to be destroyed.

**Parameter**


---

clientData	[In] The user-supplied data.
hView	[In] The HTML view.
lpURL	[In] Pointer to a string containing the URL to navigate to.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 6245

**Group**

[FR\\_HTMLEventCallbacksRec](#)**Since**[SDK LATEEST VERSION > 7.1](#)**FRHTMLEventOnDocViewNewWindow3****Syntax**

```
typedef FS_BOOL (*FRHTMLEventOnDocViewNewWindow3)(
    FS_LPVVOID clientData,
    void ppDisp,
    FS_BOOL Cancel,
    FS_DWORD dwFlags,
    FS_LPCWSTR bstrUrlContext,
    FS_LPCWSTR bstrUrl
);
```

**Description**

A callback for HTML window event handler. It is called when a new window is to be created. Go to [https://msdn.microsoft.com/en-us/library/aa768288\(VS.85\).aspx](https://msdn.microsoft.com/en-us/library/aa768288(VS.85).aspx) for more information.

**Parameter**

clientData	[In] The user-supplied data.
ppDisp	[In] An interface pointer that, optionally, receives the IDispatch interface pointer of a new WebBrowser object or an InternetExplorer object.
Cancel	[In] It determines whether the current navigation should be canceled.
dwFlags	[In] The flags from the NWMF enumeration that pertain to the new window.
bstrUrlContext	[In] The URL of the page that is opening the new window.
bstrUrl	[In] The URL that is opened in the new window.

**Return**

TRUE if the plug-in processes the event and the framework will not dispatch the event.

**Head file reference**

fr\_appExpT.h: 6264

**Group**[FR\\_HTMLEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRHTMLEventOnWillFrameClose****Syntax**

```
typedef FS_BOOL (*FRHTMLEventOnWillFrameClose)(  
    FS_LPVVOID clientData  
)
```

**Description**

A callback for HTML window event handler. It is called when the frame is closed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

TRUE if the frame can be closed, otherwise not.

**Head file reference**

fr\_appExpT.h: 6278

**Group**[FR\\_HTMLEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.2](#)**FRHTMLEventOnDispatchFun****Syntax**

```
typedef FS_BOOL (*FRHTMLEventOnDispatchFun)(  
    FS_LPVVOID clientData,  
    HWND hView,  
    FS_LPCWSTR lpwsModule,  
    FS_LPCWSTR lpwsFunName,  
    FS_LPCWSTR lpwsParam,  
    FS_WideString outRet  
)
```

**Description**

A callback for HTML window event handler. It is called when the web browser invokes the JS function to communicate with the plug-in.

**Parameter**


---

clientData	[In] The user-supplied data.
hView	[In] The HTML view.
lpwsModule	[In] The module name which the JS function is dispatched to.
lpwsFunName	[In] The function name which is invoked by the web browser.
lpwsParam	[In] The param of the JS function.
outRet	[Out] It receives the result returned to the web browser.

---

**Return**

TRUE for success, otherwise not.

**Head file reference**

fr\_appExpT.h: 6297

**Group**

[FR\\_HTMLEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

## Functions

### Functions summary

**[FRHTMLMgrGet](#)**

Gets the [FR\\_HtmlMgr](#) object is used to manage the HTML windows in *Foxit Reader*.

**[FRHTMLMgrGetActiveHtmlView](#)**

Opens a new HTML window on the specified parent window.

**[FRHTMLMgrGetHtmlViewFromHandle](#)**

Gets a new HTML window from the window handle.

**[FRHTMLMgrOpenHTMLDocument](#)**

Opens a new HTML window on the specified parent window.

**[FRHTMLMgrOpenHTMLDocument2](#)**

Opens a new HTML window on the specified parent window.

**[FRHTMLMgrOpenHTMLFromNewTab](#)**

Opens a new HTML window. Foxit Reader will create a new tab for the new HTML window.

**[FRHTMLMgrOpenHTMLFromNewTab2](#)**

Opens a new HTML window. Foxit Reader will create a new tab for the new HTML window.

**FRHTMLMgrRegisterFoxitBrowserEventHandler**

Registers the Foxit browser event handler to receive the event notifications.

**FRHTMLMgrRegisterHTMLEventHandler**

Registers the HTML event handler.

**FRHTMLMgrSetFavoritesLink**

Sets the favorite link to Foxit favorite bar.

**FRHTMLMgrSetFoxitBrowserHome**

Sets the home URL of the Foxit browser.

## Functions detail

### FRHTMLMgrGet

**Syntax**

```
FR_HTMLMgr FRHTMLMgrGet (void );
```

**Description**

Gets the [FR\\_HTMLMgr](#) object is used to manage the HTML windows in *Foxit Reader* .

**Return**

The [FR\\_HTMLMgr](#) object is used to manage the HTML windows in *Foxit Reader* .

**Head file reference**

fr\_appTempl.h: 3626

**Related method**

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

### FRHTMLMgrGetActiveHtmlView

**Syntax**

```
void* FRHTMLMgrGetActiveHtmlView (
    FR\_HTMLMgr mgr
);
```

**Description**

Opens a new HTML window on the specified parent window.

**Parameter**

---

mgr	[In] The input HTML manager object.
-----	-------------------------------------

---

**Return**

The pointer to *MFC CHtmlView* .

**Head file reference**

fr\_appTempl.h: 3678

**Related method****Since**[SDK LATEEST VERSION > 7.1](#)**FRHTMLMgrGetHtmlViewFromHandle****Syntax**

```
void* FRHTMLMgrGetHtmlViewFromHandle (
    FR_HTMLMgr mgr,
    HWND hWnd
);
```

**Description**

Gets a new HTML window from the window handle.

**Parameter**

---

mgr	[In] The input HTML manager object.
-----	-------------------------------------

---

hWnd	[In] The input HTML view handle.
------	----------------------------------

---

**Return**The pointer to *MFC CHtmlView* .**Head file reference**

fr\_appTempl.h: 3728

**Related method****Since**[SDK LATEEST VERSION > 7.2.2](#)**FRHTMLMgrOpenHTMLDocument****Syntax**

```
HWND FRHTMLMgrOpenHTMLDocument (
    FR_HTMLMgr mgr,
    FS_LPCWSTR lpURL,
    FS_LPCWSTR lpTitle,
    HWND hParentWnd
);
```

**Description**

Opens a new HTML window on the specified parent window.

**Parameter**


---

mgr	[In] The input HTML manager object.
lpURL	[In] The input URL to be opened in the HTML window.
lpTitle	[In] The title of the frame which will be created.
hParentWnd	[In] The parent window of the HTML window.

---

**Return**

The handle of the HTML window.

**Head file reference**

fr\_appTempl.h: 3664

**Related method****Since**

[SDK LATEST VERSION > 7.1](#)

**FRHTMLMgrOpenHTMLDocument2****Syntax**

```
HWND FRHTMLMgrOpenHTMLDocument2 (
    FR\_HTMLMgr mgr,
    FS\_LPCWSTR lpURL,
    FS\_LPCWSTR lpTitle,
    HWND hParentWnd,
    FS\_BOOL bHideURL
);
```

**Description**

Opens a new HTML window on the specified parent window.

**Parameter**


---

mgr	[In] The input HTML manager object.
lpURL	[In] The input URL to be opened in the HTML window.
lpTitle	[In] The title of the frame which will be created.

---

---

<b>hParentWnd</b>	[In] The parent window of the HTML window.
-------------------	--

---

<b>bHideURL</b>	[In] Whether to hide the URL or not.
-----------------	--------------------------------------

---

**Return**

The handle of the HTML window.

**Head file reference**

fr\_appTempl.h: 3758

**Related method****Since**

[SDK LATEEST VERSION > 7.3](#)

**FRHTMLMgrOpenHTMLFromNewTab****Syntax**

```
HWND FRHTMLMgrOpenHTMLFromNewTab (
    FR\_HTMLMgr mgr,
    FS\_LPCWSTR lpURL,
    FS\_LPCWSTR lpTitle,
    FS\_BOOL bCreatePanel,
    FS\_BOOL bMakeVisible,
    FS\_BOOL bAddToMRU
);
```

**Description**

Opens a new HTML window. Foxit Reader will create a new tab for the new HTML window.

**Parameter**


---

<b>mgr</b>	[In] The input HTML manager object.
------------	-------------------------------------

---

<b>lpURL</b>	[In] The input URL to be opened in the HTML window.
--------------	---

---

<b>lpTitle</b>	[In] The title of the frame which will be created.
----------------	--

---

<b>bCreatePanel</b>	[In] Whether to create the navigation panel or not.
---------------------	---

---

<b>bMakeVisible</b>	[In] Determines whether to make the HTML window visible or not.
---------------------	---

---

---

bAddToMRU	[In] Determines whether the filename is added to the most recently used (MRU) file list.
-----------	--

---

**Return**

The handle of the HTML window.

**Head file reference**

fr\_appTempl.h: 3648

**Related method****Since**

[SDK LATEEST VERSION > 7.1](#)

**FRHTMLMgrOpenHTMLFromNewTab2****Syntax**

```
HWND FRHTMLMgrOpenHTMLFromNewTab2 (
    mgr,
    lpURL,
    lpTitle,
    bCreatePanel,
    FS_BOOL bMakeVisible,
    FS_BOOL bAddToMRU,
    HICON hIcon,
    FS_BOOL bHideURL
);
```

**Description**

Opens a new HTML window. Foxit Reader will create a new tab for the new HTML window.

**Parameter**


---

mgr	[In] The input HTML manager object.
-----	-------------------------------------

---

lpURL	[In] The input URL to be opened in the HTML window.
-------	---

---

lpTitle	[In] The title of the frame which will be created.
---------	--

---

bCreatePanel	[In] Whether to create the navigation panel or not.
--------------	---

---

bMakeVisible	[In] Determines whether to make the HTML window visible or not.
--------------	---

---

bAddToMRU	[In] Determines whether the filename is added to the most recently used (MRU) file list.
-----------	--

---

---

hIcon	[In] The icon added to the tab band.
bHideURL	[In] Whether to hide the URL or not.

---

**Return**

The handle of the HTML window.

**Head file reference**

fr\_appTempl.h: 3740

**Related method****Since**

[SDK LATEEST VERSION > 7.3](#)

**FRHTMLMgrRegisterFoxitBrowserEventHandler****Syntax**

```
FS_BOOL FRHTMLMgrRegisterFoxitBrowserEventHandler (
    FR\_HTMLMgr mgr,
    FR\_FoxitBrowserEventCallbacks callbacks
);
```

**Description**

Registers the Foxit browser event handler to receive the event notifications.

**Parameter**


---

mgr	[In] The input HTML manager object.
callbacks	[In] The callback set for Foxit browser event handler.

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 3689

**Related method****Since**

[SDK LATEEST VERSION > 7.2](#)

**FRHTMLMgrRegisterHTMLEventHandler****Syntax**

```
FS_BOOL FRHTMLMgrRegisterHTMLEventHandler (
    FR\_HTMLMgr mgr,
```

---

```
FR_HTMLEventCallbacks callbacks
);
```

**Description**

Registers the HTML event handler.

**Parameter**


---

mgr	[In] The input HTML manager object.
callbacks	[In] The callback set for HTML window event handler.

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 3636

**Related method****Since**

[SDK LATEEST VERSION > 7.1](#)

**FRHTMLMgrSetFavoritesLink****Syntax**

```
void FRHTMLMgrSetFavoritesLink (
    FR_HTMLMgr mgr,
    FS_LPCWSTR lpName,
    lpURL,
    FS_LPCWSTR lpUrlIcon,
    FS_BOOL bAdd
);
```

**Description**

Sets the favorite link to Foxit favorite bar.

**Parameter**


---

mgr	[In] The input HTML manager object.
lpName	[In] The displaying name of the favorite link.
lpURL	[In] The input URL of the favorite link.

---

---

lpUrlIcon	[In] The URL of the displaying icon.
-----------	--------------------------------------

---

bAdd	[In] Whether to add the favorite link or to delete it.
------	--

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 3713

**Related method****Since**[SDK LATEEST VERSION > 7.2](#)**FRHTMLMgrSetFoxitBrowserHome****Syntax**

```
void FRHTMLMgrSetFoxitBrowserHome (
    FR HTMLMgr mgr,
    FS LPCWSTR lpURL
);
```

**Description**

Sets the home URL of the Foxit browser.

**Parameter**


---

mgr	[In] The input HTML manager object.
lpURL	[In] The input URL.

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 3701

**Related method****Since**[SDK LATEEST VERSION > 7.2](#)**FR\_Language**[Return from Used by](#)

## Description

The [FR\\_Language](#) object is used to change the language for Foxit Reader plug-in UI. You can create it by [FRLanguageCreate](#) .

### Returned from

[FRLanguageCreate](#)

### Used by

[FRLanguageChange](#)  
[FRLanguageLoadString](#)  
[FRLanguageLoadVersionRes](#)  
[FRLanguageRelease](#)  
[FRLanguageTranslateDialog](#)  
[FRLanguageTranslateDialog2](#)  
[FRLanguageTranslateMenu](#)

## Functions

### Functions summary

[FRLanguageChange](#)

Changes the language of plug-in according to the input language ID. Plug-in can change the language when the [FRAppOnLangUIChange](#) () app event occurs. You can get the current Foxit Reader language ID by [FRLanguageGetCurrentID](#) (), so that the language of plug-in can be the same as the Foxit Reader.

[FRLanguageCreate](#)

Creates a new [FR\\_Language](#) object.

[FRLanguageFormatEx](#)

Formats a translated string so that the arguments list can match the format.

[FRLanguageGetCurrentID](#)

Gets the current language ID of Foxit Reader.

[FRLanguageGetLocalLangName](#)

Gets the local language name.

[FRLanguageJSPPluginGetMessage](#)

Gets the message string for javascript plug-in.

[FRLanguageLoadString](#)

Loads the specified string. Set the *buffer* [NULL](#) to get the length of the string.

[FRLanguageLoadVersionRes](#)

Loads the file description and the legal copyright of the plug-in.

[FRLanguageModifyLogFont](#)

Modifies the attributes of a font to match the system language when the application language matches the system language, otherwise the font name will be set Tahoma as default.

[FRLanguageRelease](#)

Releases the [FR\\_Language](#) object created by [FRLanguageCreate](#) ().

[FRLanguageTranslateDialog](#)

Translates the language of dialog to another through *language* .

[FRLanguageTranslateDialog2](#)

Translates the language of dialog to another through *language* .

**FRLanguageTranslateMenu**

Translates the language of menu to another through *language* .

**Functions detail****FRLanguageChange****Syntax**

```
void FRLanguageChange (
    FR\_Language language,
    FS\_INT32 nID
);
```

**Description**

Changes the language of plug-in according to the input language ID. Plug-in can change the language when the [FRAppOnLangUIChange](#) () app event occurs. You can get the current Foxit Reader language ID by [FRLanguageGetCurrentID](#) (), so that the language of plug-in can be the same as the Foxit Reader.

**Parameter**

language	[In] The input <a href="#">FR_Language</a> object.
----------	--

nID	[In] The input language ID.
-----	-----------------------------

**Return**

void

**Head file reference**

fr\_appTempl.h: 2248

**Related method**

[FRLanguageGetCurrentID](#)

**FRLanguageCreate****Syntax**

```
FR\_Language FRLanguageCreate (
    HINSTANCE hInstance
);
```

**Description**

Creates a new [FR\\_Language](#) object.

**Parameter**

hInstance	[In] The input plug-in instance handle.
-----------	---

**Return**

The [FR\\_Language](#) object created.

**Head file reference**

fr\_appTempl.h: 2229

**FRLanguageFormatEx****Syntax**

```
void FRLanguageFormatEx (
    FS\_WideString* outFormat,
    FS\_LPCWSTR lpszFormat,
    ...
);
```

**Description**

Formats a translated string so that the arguments list can match the format.

**Parameter**

outFormat	[Out] It receives the format string.
lpszFormat	[In] Specifies a format-control string.
...	[In] arguments list.

**Return**

void

**Head file reference**

fr\_appTempl.h: 2350

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1.0](#)

**FRLanguageGetCurrentID****Syntax**

```
FS\_INT32 FRLanguageGetCurrentID (void );
```

**Description**

Gets the current language ID of Foxit Reader.

**Return**

The current language ID of Foxit Reader.

**Head file reference**

fr\_appTempl.h: 2251

**Related method**[FRLanguageChange](#)**FRLanguageGetLocalLangName****Syntax**

```
void FRLanguageGetLocalLangName (
    FS\_WideString* outName
);
```

**Description**

Gets the local language name.

**Parameter**

---

outName	[Out] It receives the local language name.
---------	--

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 2317

**Since**[SDK LATEEST VERSION > 2.1.0.4](#)**FRLanguageJSPluginGetMessage****Syntax**

```
void FRLanguageJSPluginGetMessage (
    FS\_LPCSTR lpsModuleName,
    FS\_LPCSTR lpsMessageName,
    FS\_WideString* outMessage
);
```

**Description**

Gets the message string for javascript plug-in.

**Parameter**

---

lpsModuleName	[In] The input javascript plug-in module name.
---------------	--

---

lpsMessageName	[In] The input message name.
----------------	------------------------------

---

---

outMessage	[Out] It receives the message string.
------------	---------------------------------------

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 2362

**Since**[SDK LATEST VERSION > 8.1](#)**FRLanguageLoadString****Syntax**

```
FS_INT32 FRLanguageLoadString (
    FR_Language language,
    FS_INT32 nId,
    void* buffer,
    FS_INT32 len
);
```

**Description**Loads the specified string. Set the *buffer* [NULL](#) to get the length of the string.**Parameter**


---

language	[In] The input <a href="#">FR_Language</a> object.
----------	--

---

nId	[In] Specifies the integer identifier of the string to be loaded.
-----	---

---

buffer	[Out] It receives the string loaded.
--------	--------------------------------------

---

len	[In] Specifies the length of the <i>buffer</i> in bytes.
-----	--

---

**Return**

The length of the string in bytes.

**Head file reference**

fr\_appTempl.h: 2293

**FRLanguageLoadVersionRes****Syntax**

```
void FRLanguageLoadVersionRes (
    FR_Language language,
    FS_WideString* outFileDes,
```

---

```
FS_WideString* outLegalCopyright
);
```

**Description**

Loads the file description and the legal copyright of the plug-in.

**Parameter**

language	[In] The input <a href="#">FR_Language</a> object.
outFileDes	[Out] It receives the file description.
outLegalCopyright	[Out] It receives the legal copyright.

**Return**

void

**Head file reference**

fr\_appTempl.h: 2305

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FRLanguageModifyLogFont****Syntax**

```
void FRLanguageModifyLogFont (
    LOGFONT* lf
);
```

**Description**

Modifies the attributes of a font to match the system language when the application language matches the system language, otherwise the font name will be set Tahoma as default.

**Parameter**

lf	[In/Out] The input attributes of a font to be modified.
----	---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 2339

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)**FRLanguageRelease****Syntax**

```
void FRLanguageRelease (
    FR\_Language language
);
```

**Description**

Releases the [FR\\_Language](#) object created by [FRLanguageCreate](#) ().

**Parameter**


---

language	[In] The input <a href="#">FR_Language</a> object to be released.
----------	---

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 2238

**Related method**

[FRLanguageCreate](#)

**FRLanguageTranslateDialog****Syntax**

```
void FRLanguageTranslateDialog (
    FR\_Language language,
    HWND hWnd,
    FS\_LPCWSTR pzResName
);
```

**Description**

Translates the language of dialog to another through *language* .

**Parameter**


---

language	[In] The input <a href="#">FR_Language</a> object.
----------	--

---

hWnd	[In] The input dialog handle to be translated.
------	--

---

pzResName	[In] The input name of dialog resource. The MAKEINTRESOURCE macro can be used to create this value.
-----------	--

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 2282

**FRLanguageTranslateDialog2****Syntax**

```
void FRLanguageTranslateDialog2 (
    FR\_Language language,
    HWND hWnd,
    FS\_LPCWSTR pzResName,
    FS\_BOOL bAutoAdpt
);
```

**Description**

Translates the language of dialog to another through *language* .

**Parameter**

language	[In] The input <a href="#">FR_Language</a> object.
hWnd	[In] The input dialog handle to be translated.
pzResName	[In] The input name of dialog resource. The <i>MAKEINTRESOURCE</i> macro can be used to create this value.
bAutoAdpt	[In] Whether to adapt to the DPI setting or not.

**Return**

void

**Head file reference**

fr\_appTempl.h: 2327

**FRLanguageTranslateMenu****Syntax**

```
void FRLanguageTranslateMenu (
    FR\_Language language,
    HMENU hMenu,
    FS\_LPCWSTR pzResName
);
```

**Description**

---

Translates the language of menu to another through *language* .

**Parameter**

language	[In] The input <a href="#">FR_Language</a> object.
hMenu	[In] The input menu handle to be translated.
pzResName	[In] The input name of menu resource. The <i>MAKEINTRESOURCE</i> macro can be used to create this value.

**Return**

void

**Head file reference**

fr\_appTempl.h: 2271

## FR\_Menu

### [Return from Used by](#)

#### Description

A [FR\\_Menu](#) is a menu in the Foxit Reader's menu bar. Plug-ins can create new menus, add menu items at any location in any menu, and remove menu items. Deleting an [FR\\_Menu](#) removes it from the menu bar (if it was attached) and deletes all the menu items it contains. A [FR\\_Menu](#) have a name when it is added to menu bar, plug-ins can access a menu by using [FRMenuBarGetMenuByName](#) () or [FRMenuBarGetMenuByIndex](#) (). Submenus (also called pullright menus) are [FR\\_Menu](#) objects that are attached to an [FR\\_MenuItem](#) instead of the menu bar. Your plug-in cannot directly remove a submenu. Instead, it must remove the [FR\\_MenuItem](#) to which the submenu is attached.

#### Returned from

[FRAppGetMenuBar](#)  
[FRAppGetStartMenuOfTabbedToobarMode](#)  
[FRDocViewDoPopUpMenu](#)  
[FRMenuBarGetMenuByIndex](#)  
[FRMenuBarGetMenuByName](#)  
[FRMenuItemGetParentMenu](#)  
[FRMenuItemGetSubMenu](#)  
[FRMenuItemNew](#)  
[FRMenuItemGetMenuItemByIndex](#)  
[FRMenuItemGetMenuItemByName](#)  
[FRMenuItemGetParentMenuItem](#)  
[FRMenuNew](#)  
[FRMenuNewII](#)  
[FRMenuRegisterOwnerDrawHandle](#)  
[FRMenuTrackPopup](#)

## Used by

[FRDocViewDoPopupMenu](#)  
[FRMenuBarAddMenu](#)  
[FRMenuBarDeleteMenu](#)  
[FRMenuBarGetMenuByIndex](#)  
[FRMenuBarGetMenuByName](#)  
[FRMenuBarGetMenuCount](#)  
[FRMenuBarGetMenuItemIndex](#)  
[FRMenuItemDoExecuteProc](#)  
[FRMenuItemGetClientData](#)  
[FRMenuItemGetCmdID](#)  
[FRMenuItemGetIcon](#)  
[FRMenuItemGetName](#)  
[FRMenuItemGetParentMenu](#)  
[FRMenuItemGetSubMenu](#)  
[FRMenuItemGetTitle](#)  
[FRMenuItemGetVisible](#)  
[FRMenuItemIsEnabled](#)  
[FRMenuItemIsMarked](#)  
[FRMenuItemIsSeparator](#)  
[FRMenuItemNew](#)  
[FRMenuItemRelease](#)  
[FRMenuItemSetAccelKey](#)  
[FRMenuItemSetClientData](#)  
[FRMenuItemSetComputeEnabledProc](#)  
[FRMenuItemSetComputeMarkedProc](#)  
[FRMenuItemSetDescribeText](#)  
[FRMenuItemSetExecuteProc](#)  
[FRMenuItemSetIcon](#)  
[FRMenuItemSetSubMenu](#)  
[FRMenuItemSetTitle](#)  
[FRMenuItemSetToolTip](#)  
[FRMenuItemSetVisible](#)  
[FRMenuAddMenuItem](#)  
[FRMenuDeleteMenuItem](#)  
[FRMenuDeleteOwnerDrawHandle](#)  
[FRMenuGetMenuItemByIndex](#)  
[FRMenuGetMenuItemByName](#)  
[FRMenuGetMenuItemCount](#)  
[FRMenuGetMenuItemIndex](#)  
[FRMenuGetParentMenuItem](#)  
[FRMenuGetVisible](#)  
[FRMenuRegisterOwnerDrawHandle](#)  
[FRMenuRelease](#)  
[FRMenuSetTitle](#)  
[FRMenuSetVisible](#)  
[FRMenuTrackPopup](#)

## Definitions

### Definitions summary

#### [FR\\_MENU\\_NAME\\_EDIT](#)

*Edit* menu in the menu bar

### **FR\_MENU\_NAME\_FILE**

*File* menu in the menu bar

### **FR\_MENU\_NAME\_HELP**

*Help* menu in the menu bar

### **FR\_MENU\_NAME\_TOOLS**

*Tools* menu in the menu bar

### **FR\_MENU\_NAME\_VIEW**

*View* menu in the menu bar

## Definitions detail

### **FR\_MENU\_NAME\_EDIT**

#### **Syntax**

```
#define FR_MENU_NAME_EDIT "Edit"
```

#### **Description**

*Edit* menu in the menu bar

#### **Group**

[FRMenuNames](#)

#### **Head file reference**

fr\_menuExpT.h: 85

### **FR\_MENU\_NAME\_FILE**

#### **Syntax**

```
#define FR_MENU_NAME_FILE "File"
```

#### **Description**

*File* menu in the menu bar

#### **Group**

[FRMenuNames](#)

#### **Head file reference**

fr\_menuExpT.h: 82

### **FR\_MENU\_NAME\_HELP**

#### **Syntax**

```
#define FR_MENU_NAME_HELP "Help"
```

#### **Description**

*Help* menu in the menu bar

#### **Group**

[FRMenuNames](#)

**Head file reference**

fr\_menuExpT.h: 94

**FR\_MENU\_NAME\_TOOLS****Syntax**

#define FR\_MENU\_NAME\_TOOLS "Tools"

**Description***Tools* menu in the menu bar**Group**[FRMenuNames](#)**Head file reference**

fr\_menuExpT.h: 91

**FR\_MENU\_NAME\_VIEW****Syntax**

#define FR\_MENU\_NAME\_VIEW "View"

**Description***View* menu in the menu bar**Group**[FRMenuNames](#)**Head file reference**

fr\_menuExpT.h: 88

## Functions

### Functions summary

**[FRMenuAddMenuItem](#)**

Inserts a menu item to the specified menu at the specified location. If *iIndex* is -1 or is greater than ([FRMenuGetMenuItemCount](#) ()-1), the *menuitem* will be appended to the menu.

**[FRMenuCloseActivePopupMenu](#)**

Closes the active pop-up menu.

**[FRMenuDeleteMenuItem](#)**

Removes a menu item from specified menu, then destroys the menu item.

**[FRMenuDeleteOwnerDrawHandle](#)**

Releases the menu owner-draw handler from [FRMenuRegisterOwnerDrawHandle](#) .

**[FRMenuGetMenuItemByIndex](#)**

Gets the menu item at the specified location in the specified menu.

**[FRMenuGetMenuItemByName](#)**

Gets the menu item by the specified name in the specified menu.

**[FRMenuGetMenuItemCount](#)**

Gets the number of item in the specified menu.

**[FRMenuGetMenuItemIndex](#)**

Gets the index of the specified menu item in the specified menu.

**[FRMenuGetParentMenuItem](#)**

Gets the parent menu item for the specified menu.

**[FRMenuGetVisible](#)**

Checks whether the menu is visible or not.

**[FRMenuNew](#)**

Creates a new menu.

**[FRMenuNewII](#)**

Creates a new menu from a menu handle.

**[FRMenuRegisterOwnerDrawHandle](#)**

Registers a menu owner-draw handler. It is proper for a sub-menu.

**[FRMenuRelease](#)**

Removes the specified menu and releases it.

**[FRMenuSetTitle](#)**

Sets the title of the menu.

**[FRMenuSetVisible](#)**

Sets the menu visible or not.

**[FRMenuTrackPopup](#)**

Displays a floating pop-up menu at the specified location and tracks the selection of items on the pop-up menu. You can call [FRMenuItemDoExecuteProc](#) () to make the [FRExecuteProc](#) () callback invoked.

**[FRMenuTrackPopupMenu](#)**

The returned command ID.

## Functions detail

### [FRMenuAddMenuItem](#)

**Syntax**

```
FS_BOOL FRMenuAddMenuItem (
    FR\_Menu menu,
    FR\_MenuItem menuitem,
    FS\_INT32 iIndex
);
```

**Description**

Inserts a menu item to the specified menu at the specified location. If *iIndex* is -1 or is greater than ([FRMenuGetMenuItemCount](#) ()-1), the *menuitem* will be appended to the menu.

**Parameter**

---

menu	[In] The menu into which the menu item is added.
------	--

---

menuitem	[In] Then menu item to add.
----------	-----------------------------

---

---

iIndex	[In] The inserted location.
--------	-----------------------------

---

**Return**

Return [TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_menuTempl.h: 174

**Related method**

[FRMenuCloseActivePopupMenu](#)

**FRMenuCloseActivePopupMenu**

**Syntax**

FS\_BOOL FRMenuCloseActivePopupMenu (void );

**Description**

Closes the active pop-up menu.

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_menuTempl.h: 305

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRMenuDeleteMenuItem**

**Syntax**

```
FS_BOOL FRMenuDeleteMenuItem (
    FR\_Menu menu,
    FR\_MenuItem menuitem
);
```

**Description**

Removes a menu item from specified menu, then destroys the menu item.

**Parameter**

---

menu	[In] The menu for which the menu item is deleted.
------	---

---

---

menuitem	[In] The menu item to delete.
----------	-------------------------------

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_menuTempl.h: 182

**Related method**

[FRMenuAddMenuItem](#)

**FRMenuDeleteOwnerDrawHandle****Syntax**

```
void FRMenuDeleteOwnerDrawHandle (
    FR\_MenuOwnerDrawHandler menuHandle
);
```

**Description**

Releases the menu owner-draw handler from [FRMenuRegisterOwnerDrawHandle](#).

**Parameter**

---

menuHandle	[In] The menu owner-draw handler.
------------	-----------------------------------

---

**Return**

void

**Head file reference**

fr\_menuTempl.h: 209

**Related method**

[FRMenuRegisterOwnerDrawHandle](#)

**FRMenuGetMenuItemByIndex****Syntax**

```
FR_MenuItem FRMenuGetMenuItemByIndex (
    FR\_Menu menu,
    FS\_INT32 iIndex
);
```

**Description**

Gets the menu item at the specified location in the specified menu.

**Parameter**

---

menu	[In] The menu whose item is obtained.
------	---------------------------------------

---

---

iIndex	[In] The index of the menu item in <i>menu</i> to obtained. The index range is 0 to ( <a href="#">FRMenuGetMenuItemCount</a> -1).
--------	---

---

**Return**

The specified menu item. It returns [NULL](#) if *menu* is [NULL](#) , if the index is less than zero, or the index is greater than the number of menu items in the menu.

**Head file reference**

fr\_menuTempl.h: 130

**Related method**

[FRMenuGetParentMenuItem](#)

[FRMenuGetMenuItemByName](#)

[FRMenuGetMenuItemCount](#)

**FRMenuGetMenuItemByName****Syntax**

```
FR_MenuItem FRMenuGetMenuItemByName (
    FR\_Menu menu,
    FS\_LPCSTR csItemName
);
```

**Description**

Gets the menu item by the specified name in the specified menu.

**Parameter**


---

menu	[In] The menu whose item is obtained.
------	---------------------------------------

---

csItemName	[In] The menu item name.
------------	--------------------------

---

**Return**

The specified menu item. It returns [NULL](#) if *menu* is [NULL](#) , if the named menu item is not exist.

**Head file reference**

fr\_menuTempl.h: 131

**Related method**

[FRMenuGetParentMenuItem](#)

[FRMenuGetMenuItemByIndex](#)

**FRMenuGetMenuItemCount****Syntax**

```
FS_INT32 FRMenuGetMenuItemCount (
```

```
FR_Menu menu  
);
```

**Description**

Gets the number of item in the specified menu.

**Parameter**

---

menu	[In] The menu whose number of items is obtained.
------	--

---

**Return**

The number of items in the specified menu.

**Head file reference**

fr\_menuTempl.h: 140

**Related method**

[FRMenuGetMenuItemByIndex](#)

**FRMenuGetMenuItemIndex****Syntax**

```
FS_INT32 FRMenuGetMenuItemIndex (  
    FR_Menu menu,  
    FR_MenuItem menuitem  
,
```

**Description**

Gets the index of the specified menu item in the specified menu.

**Parameter**

---

menu	[In] The menu in which <i>menuitem</i> is located.
------	--

---

---

menuitem	[In] The menu item whose index is obtained.
----------	---

---

**Return**

The index of menu item. Or -1 if *menuitem* is invalid.

**Head file reference**

fr\_menuTempl.h: 198

**Related method**

[FRMenuGetMenuItemByIndex](#)

**FRMenuGetParentMenuItem**

**Syntax**

```
FR_MenuItem FRMenuGetParentMenuItem (
    FR\_Menu menu
);
```

**Description**

Gets the parent menu item for the specified menu.

**Parameter**

---

menu	[In] The menu whose parent menu item is obtained.
------	---

---

**Return**

The parent menu item for which the specified menu is a submenu. [NULL](#) if the specified menu is not a submenu.

**Head file reference**

fr\_menuTempl.h: 124

**Related method**

[FRMenuGetMenuItemByIndex](#)  
[FRMenuGetMenuItemByName](#)

**FRMenuGetVisible****Syntax**

```
FS_BOOL FRMenuGetVisible (
    FR\_Menu menu
);
```

**Description**

Checks whether the menu is visible or not.

**Parameter**

---

menu	[In] The input menu object.
------	-----------------------------

---

**Return**

[TRUE](#) if the menu is visible, otherwise not.

**Head file reference**

fr\_menuTempl.h: 256

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRMenuNew

### Syntax

```
FR_Menu FRMenuNew (void );
```

### Description

Creates a new menu.

### Return

The newly created menu.

### Head file reference

fr\_menuTempl.h: 103

### Related method

[FRMenuRelease](#)

[FRMenuBarGetMenuByName](#)

[FRMenuBarAddMenu](#)

[FRMenuBarDeleteMenu](#)

## FRMenuNewII

### Syntax

```
FR_Menu FRMenuNewII (
    HMENU hMenu
);
```

### Description

Creates a new menu from a menu handle.

### Parameter

---

hMenu	[In] The input menu handle.
-------	-----------------------------

---

### Return

The newly created menu.

### Head file reference

fr\_menuTempl.h: 279

### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRMenuRegisterOwnerDrawHandle

### Syntax

```
FR_MenuOwnerDrawHandler FRMenuRegisterOwnerDrawHandle (
```

```
    FR\_Menu menu,  
    FR\_MenuOwnerDrawCallbacksRec callbacks  
);
```

**Description**

Registers a menu owner-draw handler. It is proper for a sub-menu.

**Parameter**

---

menu	[In] The input menu.
callbacks	[In] The callback set for menu owner-draw handler.

---

**Return**

The menu owner-draw handler.

**Head file reference**

fr\_menuTempl.h: 210

**Related method**

[FRMenuDeleteOwnerDrawHandle](#)

**FRMenuRelease****Syntax**

```
void FRMenuRelease (  
    FR\_Menu menu  
);
```

**Description**

Removes the specified menu and releases it.

**Parameter**

---

menu	[In] The menu to be released.
------	-------------------------------

---

**Return****Head file reference**

fr\_menuTempl.h: 89

**FRMenuSetTitle****Syntax**

```
void FRMenuSetTitle (  
    FR\_Menu menu,
```

```
FS_LPCWSTR lpwsTitle  
);
```

**Description**

Sets the title of the menu.

**Parameter**

menu	[In] The input menu object.
lpwsTitle	[In] The title of the menu.

**Return**

void

**Head file reference**

fr\_menuTempl.h: 267

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRMenuSetVisible****Syntax**

```
void FRMenuSetVisible (  
    FR_Menu menu,  
    FS_BOOL bShow  
,
```

**Description**

Sets the menu visible or not.

**Parameter**

menu	[In] The input menu object.
bShow	[In] Indicates whether sets the menu visible or not.

**Return**

void

**Head file reference**

fr\_menuTempl.h: 244

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRMenuItemTrackPopup****Syntax**

```
FR_MenuItem FRMenuItemTrackPopup (
    FR\_Menu menu,
    FS\_INT32 x,
    FS\_INT32 y
);
```

**Description**

Displays a floating pop-up menu at the specified location and tracks the selection of items on the pop-up menu. You can call [FRMenuItemDoExecuteProc](#) () to make the [FRExecuteProc](#) () callback invoked.

**Parameter**

menu	[In] The menu to be displayed.
x	[In] The x-coordinate of the upper-left corner of the menu.
y	[In] The y-coordinate of the upper-left corner of the menu.

**Return**

The menu item you selected in the pop-up menu.

**Head file reference**

fr\_menuTempl.h: 230

**Related method**
[FRDocViewDoPopUpMenu](#)
**FRMenuItemTrackPopupMenu****Syntax**

```
FS_UINT FRMenuItemTrackPopupMenu (
    HMENU hMenu,
    FS\_INT32 x,
    FS\_INT32 y,
    FS\_BOOL bReturnCmdID,
    HWND hOwner,
    FRGetMessageStringProc pFunProc,
    FS\_BOOL bRightAlign
);
```

**Description**

The returned command ID.

**Parameter**

hMenu	[In] The input menu handle.
x	[In] The x pos.
y	[In] The y pos.
bReturnCmdID	[In] Whether to return the command ID or not.
hOwner	[In] The parent window.
pFunProc	[In] The input callback function to receive the tooltip.
bRightAlign	[In] Whether to align to right or not.

**Return**

Pops up a menu.

**Head file reference**

fr\_menuTempl.h: 289

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2](#)

## FR\_MenuBar

[Return from](#) [Used by](#)

**Description**

The [FR\\_MenuBar](#) is the Foxit Reader menu bar and contains a list of all menus. There is only one menu bar within Foxit Reader. Plug-ins can add new menus to, or remove any menu from, the menu bar. The menu bar can be hidden from the user's view.

**Returned from**

[FRApGetMenuBar](#)

## Used by

[FRMenuBarAddMenu](#)  
[FRMenuBarDeleteMenu](#)  
[FRMenuBarGetMenuByIndex](#)  
[FRMenuBarGetMenuByName](#)  
[FRMenuBarGetMenuCount](#)  
[FRMenuBarGetMenuItemIndex](#)

## Functions

### Functions summary

#### [FRMenuBarAddMenu](#)

Inserts a menu into a menu bar. It does nothing if the *menuBar* or *menu* is [NULL](#).

#### [FRMenuBarDeleteMenu](#)

Removes the menu from menu bar, then destroy the menu.

#### [FRMenuBarGetMenuByIndex](#)

Gets the menu with the specified index.

#### [FRMenuBarGetMenuByName](#)

Gets the menu with the specified name.

#### [FRMenuBarGetMenuCount](#)

Gets the number of menus in menu bar.

#### [FRMenuBarGetMenuItemIndex](#)

Gets the index of specified menu bar.

### Functions detail

#### FRMenuBarAddMenu

##### Syntax

```
FS_BOOL FRMenuBarAddMenu (
    FR\_MenuBar menuBar,
    FR\_Menu menu,
    FS\_LPCWSTR szItemTitle,
    FS\_LPCSTR szItemName,
    FS\_INT32 iIndex
);
```

##### Description

Inserts a menu into a menu bar. It does nothing if the *menuBar* or *menu* is [NULL](#).

##### Parameter

---

menuBar	[In] The menu bar into which the menu is added.
---------	---

---

menu	[In] The menu to add.
------	-----------------------

---

szItemTitle	[In] The title of the menu to display on menu bar.
-------------	--

---

---

szItemName	[In] The name to indicate the menu.
------------	-------------------------------------

---

iIndex	[In] The inserted location. The range of <i>iIndex</i> is 0 to <a href="#">FRMenuBarGetMenuCount ()</a> .
--------	---

---

**Return**

[TRUE](#) for success. It returns [FALSE](#) if *menu* is [NULL](#) or the *szItemName* has existed.

**Head file reference**

fr\_menuTempl.h: 56

**Related method**

[FRMenuBarDeleteMenu](#)

**Note:**It is strongly encouraged that you begin your menu names with the plug-in name (separated by a colon) to avoid name collisions when more than one plug-in is present. For example, if my plug-in is named myPlug, it might add a menu whose name is myPlug:Options.

**FRMenuBarDeleteMenu****Syntax**

```
FS_BOOL FRMenuBarDeleteMenu (
    FR\_MenuBar menuBar,
    FR\_Menu menu
);
```

**Description**

Removes the menu from menu bar, then destroy the menu.

**Parameter**


---

menuBar	[In] The menu bar whose menu is deleted.
---------	--

---

menu	[In] The menu to delete.
------	--------------------------

---

**Return**

[TRUE](#) for success, [FALSE](#) if *menuBar* or *menu* is [NULL](#) .

**Head file reference**

fr\_menuTempl.h: 68

**Related method**

[FRMenuBarAddMenu](#)

**Note:**This method will destroy the specified menu, don't call FRMenuRelease to destroy the menu again.

## FRMenuBarGetMenuByIndex

### Syntax

```
FR_Menu FRMenuBarGetMenuByIndex (  
    FRMenuBar menuBar,  
    FS\_INT32 iIndex  
);
```

### Description

Gets the menu with the specified index.

### Parameter

---

menuBar	[In] The menu bar for which the menu is located.
---------	--

---

iIndex	[In] The index of the menu to obtain. The range of <i>iIndex</i> is 0 to ( <i>GetMenuCount()</i> );
--------	---

---

### Return

The menu with the specified index. It returns [NULL](#) if there is no such menu or if *menuBar* is [NULL](#).

### Head file reference

[fr\\_menuTempl.h: 26](#)

### Related method

[FRMenuBarGetMenuByName](#)

[FRMenuBarGetMenuItem](#)

## FRMenuBarGetMenuByName

### Syntax

```
FR_Menu FRMenuBarGetMenuByName (  
    FRMenuBar menuBar,  
    FS\_LPCSTR szName  
);
```

### Description

Gets the menu with the specified name.

### Parameter

---

menuBar	[In] The menu bar for which the menu is located.
---------	--

---

szName	[In] The menu name.
--------	---------------------

---

**Return**

The menu with the specified name. It returns [NULL](#) if there is no such menu or if *menuBar* is [NULL](#).

**Head file reference**

fr\_menuTempl.h: 38

**Related method**

[FRMenuBarGetMenuByIndex](#)

**FRMenuBarGetMenuCount****Syntax**

```
FS_INT32 FRMenuBarGetMenuCount (
    FR\_MenuBar menuBar
);
```

**Description**

Gets the number of menus in menu bar.

**Parameter**

---

menuBar	[In] The menu bar for which the number of menus is obtained.
---------	--

---

**Return**

The number of menus in the menu bar, not including submenus. Return 0 if *menuBar* is [NULL](#).

**Head file reference**

fr\_menuTempl.h: 21

**Related method**

[FRMenuBarGetMenuByIndex](#)

**FRMenuBarGetMenuItemIndex****Syntax**

```
FS_INT32 FRMenuBarGetMenuItemIndex (
    FR\_MenuBar menuBar,
    FR\_Menu menu
);
```

**Description**

Gets the index of specified menu bar.

**Parameter**

---

menuBar	[In] The menu bar whose menu's index is obtained.
menu	[In] The menu whose index is obtained.

---

**Return**

The index of specified menu. The range of the return value is 0 to ([FRMenuBarGetMenuCount](#) -1).

**Head file reference**

fr\_menuTempl.h: 39

**Related method**

[FRMenuBarGetMenuItemByIndex](#)

## FR\_MenuItem

### [Return from Used by](#)

#### Description

A [FR\\_MenuItem](#) is a menu item in a menu. It has attributes, including the following:

- A name that indicates the menu item.
- A title that displayed on screen.
- A procedure to execute when the menu item is selected.
- A procedure to compute whether the menu item is enabled.
- A procedure to compute whether the menu item is checkmarked, and whether it has a sub-menu.

A plug-in can simulate a user selecting a menu item by calling [FRMenuDoExecuteProc](#) (). If the menu item is disabled, [FRMenuDoExecuteProc](#) () returns without doing anything. [FRMenuDoExecuteProc](#) () works even when the menu item is not displayed (for example, if it has not been added to a menu, its menu is not displayed, or the menu bar is not visible). Plug-ins can set all attributes of menu items they create, but must not set the Execute procedure of the Foxit Reader built-in menu items.

You are strongly encouraged to begin your plug-in's menu item names with your plug-in's name (separated by a colon) to avoid name collisions when more than one plug-in is present. For example, if your plug-in is named *myPlugin* , it ight add menu items whose names are *myPlugin:Close* and *myPlugin:Save* .

#### Returned from

[FRDocViewDoPopUpMenu](#)  
[FRMenuItemGetMenuItemByIndex](#)  
[FRMenuItemGetMenuItemByName](#)  
[FRMenuItemGetParentMenuItem](#)  
[FRMenuItemTrackPopup](#)  
[FRMenuItemNew](#)

#### Used by

[\*\*FRMenuItemAddMenuItem\*\*](#)  
[\*\*FRMenuItemDeleteMenuItem\*\*](#)  
[\*\*FRMenuItemGetMenuItemIndex\*\*](#)  
[\*\*FRMenuItemDoExecuteProc\*\*](#)  
[\*\*FRMenuItemGetClientData\*\*](#)  
[\*\*FRMenuItemGetCmdID\*\*](#)  
[\*\*FRMenuItemGetIcon\*\*](#)  
[\*\*FRMenuItemGetName\*\*](#)  
[\*\*FRMenuItemGetParentMenu\*\*](#)  
[\*\*FRMenuItemGetSubMenu\*\*](#)  
[\*\*FRMenuItemGetTitle\*\*](#)  
[\*\*FRMenuItemGetVisible\*\*](#)  
[\*\*FRMenuItemIsEnabled\*\*](#)  
[\*\*FRMenuItemIsMarked\*\*](#)  
[\*\*FRMenuItemIsSeparator\*\*](#)  
[\*\*FRMenuItemRelease\*\*](#)  
[\*\*FRMenuItemSetAccelKey\*\*](#)  
[\*\*FRMenuItemSetClientData\*\*](#)  
[\*\*FRMenuItemSetComputeEnabledProc\*\*](#)  
[\*\*FRMenuItemSetComputeMarkedProc\*\*](#)  
[\*\*FRMenuItemSetDescribeText\*\*](#)  
[\*\*FRMenuItemSetExecuteProc\*\*](#)  
[\*\*FRMenuItemSetIcon\*\*](#)  
[\*\*FRMenuItemSetSubMenu\*\*](#)  
[\*\*FRMenuItemSetTitle\*\*](#)  
[\*\*FRMenuItemSetToolTip\*\*](#)  
[\*\*FRMenuItemSetVisible\*\*](#)

## Functions

### Functions summary

#### [\*\*FRMenuItemDoExecuteProc\*\*](#)

Executes a menu item's [\*\*FRExecuteProc\*\*](#) () .

#### [\*\*FRMenuItemGetClientData\*\*](#)

Gets the user-supplied data structure that set to menu item using [\*\*FRMenuItemSetClientData\*\*](#) () .

#### [\*\*FRMenuItemGetCmdID\*\*](#)

Gets the cmd ID of the menu item.

#### [\*\*FRMenuItemGetIcon\*\*](#)

Gets the icon of the *menuitem* .

#### [\*\*FRMenuItemGetName\*\*](#)

Gets the name of the specified menu item.

#### [\*\*FRMenuItemGetParentMenu\*\*](#)

Gets the menu in which the specified menu item appears.

#### [\*\*FRMenuItemGetSubMenu\*\*](#)

Gets the submenu of a menu item.

#### [\*\*FRMenuItemGetTitle\*\*](#)

Gets a menu item's title, which is the string that displayed in user interface.

#### [\*\*FRMenuItemGetVisible\*\*](#)

Checks whether the menu item is visible or not.

#### [\*\*FRMenuItemIsEnabled\*\*](#)

Tests whether the specified menu item is enabled.

**FRMenuItemIsMarked**

Tests whether the specified menu item is marked.

**FRMenuItemIsSeparator**

Tests whether the specified menu item is a separator or a normal menu item.

**FRMenuItemNew**

Creates a new menu item.

**FRMenuItemRelease**

Removes the specified menu item and releases it.

**FRMenuItemSetAccelKey**

Sets the hot key to a exist menu item.

**FRMenuItemSetClientData**

Sets the user-supplied data structure which is passed to [FRExecuteProc](#) , [FRComputeEnabledProc](#) , [FRComputeMarkedProc](#) , [FRFreeDataProc](#) .

**FRMenuItemSetComputeEnabledProc**

Sets the user-supplied procedure to call to determine whether the menu item is enabled.

**FRMenuItemSetComputeMarkedProc**

Sets the user-supplied procedure to call to determine whether the menu item is marked.

**FRMenuItemSetDescribeText**

Sets the describe text to a menu item. The description text is shown in the right of the status bar while the mouse over the specified menu item.

**FRMenuItemSetExecuteProc**

Sets the user-supplied procedure to execute whenever the menu item is chosen. Client must not set the procedure of the Foxit Reader's build-in menu items.

**FRMenuItemSetIcon**

Sets the icon of the menu item.

**FRMenuItemSetSubMenu**

Attaches a sub-menu to a menu item.

**FRMenuItemSetTitle**

Sets a menu item's title.

**FRMenuItemSetToolTip**

Sets the tool tip that appears when the mouse over the item area.

**FRMenuItemSetVisible**

Whether sets the menu item visible or not.

## Functions detail

### FRMenuItemDoExecuteProc

**Description**

Executes a menu item's [FRExecuteProc](#) () .

**Parameter**

---

menuitem	[In] The menu item to execute.
----------	--------------------------------

---

**Return**

void

**Head file reference**

fr\_menuTempl.h: 232

**Related method**

[FRMenuItemSetExecuteProc](#)

**Note:** You cannot execute a menu item that has a sub-menu.

**FRMenuItemGetClientData****Syntax**

```
void* FRMenuItemGetClientData (
    FR\_MenuItem menuitem
);
```

**Description**

Gets the user-supplied data structure that set to menu item using [FRMenuItemSetClientData](#) ().

**Parameter**

---

menuitem	[In] The menu item whose user-supplied data is obtained.
----------	--

---

**Return**

A pointer to a user-supplied data structure. The data structure should contain three types user-supplied data for [FRExecuteProc](#) (), [FRComputeEnabledProc](#) (), [FRComputeMarkedProc](#) (). It returns [NULL](#) if no client data to be set.

**Head file reference**

fr\_menuTempl.h: 550

**Related method**

[FRMenuItemSetClientData](#)

**FRMenuItemGetCmdID****Syntax**

```
FS_INT32 FRMenuItemGetCmdID (
    FR\_MenuItem menuitem
);
```

**Description**

Gets the cmd ID of the menu item.

**Parameter**

---

menuitem	[In] The input menu item.
----------	---------------------------

---

**Return**

The cmd ID of the menu item.

**Head file reference**

fr\_menuTempl.h: 576

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRMenuItemGetIcon****Syntax**

```
FS_DIBitmap FRMenuItemGetIcon (
    FR MenuItem menuitem
);
```

**Description**

Gets the icon of the *menuitem*.

**Parameter**

---

menuitem	[In] The menu item whose icon is obtained.
----------	--

---

**Return**

The [FS\\_DIBitmap](#) object that coorespond a menu item icon, or [NULL](#) if the menu item does not have a icon.

**Head file reference**

fr\_menuTempl.h: 383

**Related method**

[FRMenuItemSetIcon](#)

**FRMenuItemGetName****Syntax**

```
FS_BOOL FRMenuItemGetName (
    FR MenuItem menuitem,
    FS ByteString* outName
);
```

**Description**

Gets the name of the specified menu item.

#### Parameter

---

menuitem	[In] The menu item whose name is obtained.
outName	[Out] (Filled by this method) The char buffer to receive the menu item's name.

---

#### Return

[TRUE](#) if *outName* is filled success, [FALSE](#) otherwise.

#### Head file reference

fr\_menuTempl.h: 426

#### Related method

[FRMenuItemGetMenuItemByName](#)

## FRMenuItemGetParentMenu

#### Syntax

```
FR_Menu FRMenuItemGetParentMenu (
    FR\_MenuItem menuitem
);
```

#### Description

Gets the menu in which the specified menu item appears.

#### Parameter

---

menuitem	[In] The menu item whose parent menu is obtained.
----------	---

---

#### Return

The menu in which the specified menu item appears. It returns [NULL](#) if this menu item is not in a menu.

#### Head file reference

fr\_menuTempl.h: 373

## FRMenuItemGetSubMenu

#### Syntax

```
FR_Menu FRMenuItemGetSubMenu (
    FR\_MenuItem menuitem
);
```

**Description**

Gets the submenu of a menu item.

**Parameter**


---

menuitem	[In] The menu item whose submenu is obtained.
----------	---

---

**Return**

The submenu or [NULL](#) if the *menuitem* does not have a sub-menu.

**Head file reference**

fr\_menuTempl.h: 352

**Related method**

[FRMenuItemSetSubMenu](#)

**FRMenuItemGetTitle****Syntax**

```
FS_BOOL FRMenuItemGetTitle (
    FR MenuItem menuitem,
    FS WideString* outTitle
);
```

**Description**

Gets a menu item's title, which is the string that displayed in user interface.

**Parameter**


---

menuitem	[In] The menu item whose title is obtained.
----------	---

---

outTitle	[Out] (Filled by this method) A unicode string buffer to receive the menu item's title.
----------	---

---

**Return**

[TRUE](#) if *outTitle* is filled success, [FALSE](#) otherwise.

**Head file reference**

fr\_menuTempl.h: 404

**Related method**

[FRMenuItemSetTitle](#)

**FRMenuItemGetVisible****Syntax**

```
FS_BOOL FRMenuItemGetVisible (
```



```
FR_MenuItem menuitem  
);
```

**Description**

Checks whether the menu item is visible or not.

**Parameter**

menuitem	[In] The input menu item.
----------	---------------------------

**Return**

[TRUE](#) if the menu item is visible, otherwise not.

**Head file reference**

fr\_menuTempl.h: 616

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRMenuItemIsEnabled****Syntax**

```
FS_BOOL FRMenuItemIsEnabled (  
    FR_MenuItem menuitem  
) ;
```

**Description**

Tests whether the specified menu item is enabled.

**Parameter**

menuitem	[In] The menu item whose enabled flag is obtained.
----------	--

**Return**

[TRUE](#) if *menuitem* is enabled, if *menuitem* is [NULL](#), or if *menuitem* has no [FRComputeEnabledProc](#) (). It returns [FALSE](#) if the menu item is disabled or its [FRComputeEnabledProc](#) () raise an exception.

**Head file reference**

fr\_menuTempl.h: 493

**Related method**

[FRMenuItemSetComputeEnabledProc](#)

**FRMenuItemIsMarked**

**Syntax**

```
BOOL FRMenuItemIsMarked (
    FR MenuItem menuitem
);
```

**Description**

Tests whether the specified menu item is marked.

**Parameter**

---

menuitem	[In] The menu item whose mark flag is obtained.
----------	---

---

**Return**

[TRUE](#) if *menuitem* is marked. It returns [FALSE](#) if *menuitem* is [NULL](#). if the menu item dose not have a [FRComputeMarkedProc](#) () or if it raise an exception.

**Head file reference**

fr\_menuTempl.h: 509

**Related method**

[FRMenuItemSetComputeMarkedProc](#)

**FRMenuItemIsSeparator****Syntax**

```
FS_BOOL FRMenuItemIsSeparator (
    FR MenuItem menuitem
);
```

**Description**

Tests whether the specified menu item is a separator or a normal menu item.

**Parameter**

---

menuitem	[In] The menu item to test.
----------	-----------------------------

---

**Return**

[TRUE](#) if a menu item is a separator, [FALSE](#) otherwise.

**Head file reference**

fr\_menuTempl.h: 460

**FRMenuItemNew****Syntax**

```
FR_MenuItem FRMenuItemNew (
    FS LPCSTR szName,
```

---

```
FS_LPCWSTR wszTitle,
FS_DIBitmap bmp,
FS_BOOL bSeparator,
FR_Menu submenu
);
```

**Description**

Creates a new menu item.

**Parameter**

szName	[In] The name of the menu item to create.
wszTitle	[In] The title to display for this menu item.
bmp	[In] The icon to show in the menu item. or <u>NULL</u> if no icon is shown. The size of icon is 24 * 24 sample bitmap.
bSeparator	[In] A flag that indicate whether the menu item is a separator. If <u>TRUE</u> , the new menu item is a separator used to leave space between groups of related menu items. If <u>FALSE</u> , the menu item is a normal item. The <i>szName</i> and <i>wszTitle</i> are both ignored when a menu item is a separator.
submenu	[In] A submenu (if any) for which this menu item is parent. It can be <u>NULL</u> if this menu item does not have a sub-menu.

**Return**

The newly created menu item.

**Head file reference**

fr\_menuTempl.h: 322

**Related method**

[FRMenuItemRelease](#)

[FRMenuAddMenuItem](#)

**FRMenuItemRelease****Syntax**

```
void FRMenuItemRelease (
    FR_MenuItem menuitem
);
```

**Description**

Removes the specified menu item and releases it.

**Parameter**

---

menuitem	[In] The menu item to release.
----------	--------------------------------

---

**Return**

void

**Head file reference**

fr\_menuTempl.h: 336

**Related method**[FRMenuItemNew](#)**FRMenuItemSetAccelKey****Syntax**

```
FS_BOOL FRMenuItemSetAccelKey (
    FR\_MenuItem menuitem,
    FS_BOOL bAlt,
    FS_BOOL bShift,
    FS_BOOL bCtrl,
    FS_CHAR key
);
```

**Description**

Sets the hot key to a exist menu item.

**Parameter**

---

menuitem	[In] The menu item whose hot key will be set.
----------	---

---

---

bAlt	[In] A flag indicate the Alt key.
------	-----------------------------------

---

---

bShift	[In] A flag indicate the Shift key.
--------	-------------------------------------

---

---

bCtrl	[In] A flag indicate the Ctrl key.
-------	------------------------------------

---

---

key	[In] The hot key.
-----	-------------------

---

**Return**[TRUE](#) means successful, otherwise not.**Head file reference**

fr\_menuTempl.h: 562

**Related method****FRMenuItemSetClientData****Syntax**

```
FS_BOOL FRMenuItemSetClientData (
    FR MenuItem menuitem,
    void* pClientData,
    FRFreeDataProc callback
);
```

**Description**

Sets the user-supplied data structure which is passed to [FRExecuteProc](#), [FRComputeEnabledProc](#), [FRComputeMarkedProc](#), [FRFreeDataProc](#).

**Parameter**

menuitem	[In] The menu item whose user-supplied data is set.
pClientData	[In] A pointer to a user-supplied data structure. The data structure should contain three types user-supplied data for <a href="#">FRExecuteProc</a> (), <a href="#">FRComputeEnabledProc</a> (), <a href="#">FRComputeMarkedProc</a> ().
callback	[In] It is called when Foxit Reader will free objects such as the menu item.

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_menuTempl.h: 475

**Related method**

[FRMenuItemSetExecuteProc](#)  
[FRMenuItemSetComputeEnabledProc](#)  
[FRMenuItemSetComputeMarkedProc](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRMenuItemSetComputeEnabledProc****Syntax**

```
FS_BOOL FRMenuItemSetComputeEnabledProc (
    FR MenuItem menuitem,
    FRComputeEnabledProc proc
```

);

**Description**

Sets the user-supplied procedure to call to determine whether the menu item is enabled.

**Parameter**

menuitem	[In] The menu item whose <a href="#">FRComputeEnabledProc</a> is set.
proc	[In] A user-supplied callback to call whenever the Foxit Reader needs to know whether <i>menuitem</i> should be enabled.

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_menuTempl.h: 478

**Related method**

[FRMenuItemIsEnabled](#)

[FRMenuItemSetClientData](#)

[FRMenuItemSetExecuteProc](#)

[FRMenuItemSetComputeMarkedProc](#)

**Note:** If a user-supplied data need to pass to proc, using FRMenuItemSetClientData() to set the user-supplied data first.

## FRMenuItemSetComputeMarkedProc

**Syntax**

```
FS_BOOL FRMenuItemSetComputeMarkedProc (
    FR\_MenuItem menuitem,
    FRComputeMarkedProc proc
);
```

**Description**

Sets the user-supplied procedure to call to determine whether the menu item is marked.

**Parameter**

menuitem	[In] The menu item whose <a href="#">FRComputeMarkedProc</a> is set.
proc	[In] A user-supplied callback to call whenever the Foxit Reader needs to know whether <i>menuitem</i> should be marked.

**Return**

[TRUE](#) means successful, otherwise not.

#### Head file reference

fr\_menuTempl.h: 479

#### Related method

[FRMenuItemIsMarked](#)

[FRMenuItemSetClientData](#)

[FRMenuItemSetExecuteProc](#)

[FRMenuItemSetComputeEnabledProc](#)

**Note:** If a user-supplied data need to pass to proc, using FRMenuItemSetClientData() to set the user-supplied data first.

### FRMenuItemSetDescribeText

#### Syntax

```
void FRMenuItemSetDescribeText (
    FR_MenuItem menuitem,
    FS_LPCWSTR wszText
);
```

#### Description

Sets the describe text to a menu item. The description text is shown in the right of the status bar while the mouse over the specified menu item.

#### Parameter

menuitem	[In] The menu item whose describe text is set.
----------	--

wszText	[In] The describe text.
---------	-------------------------

#### Return

void

#### Head file reference

fr\_menuTempl.h: 443

#### Related method

[FRMenuItemSetToolTip](#)

### FRMenuItemSetExecuteProc

#### Syntax

```
FS_BOOL FRMenuItemSetExecuteProc (
    FR_MenuItem menuitem,
    FRExecuteProc proc
```

);

**Description**

Sets the user-supplied procedure to execute whenever the menu item is chosen. Client must not set the procedure of the Foxit Reader's build-in menu items.

**Parameter**

menuitem	[In] The menu item whose procedure is set.
proc	[In] A user-supplied callback to call whenever <i>menuitem</i> is selected.

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_menuTempl.h: 469

**Related method**

[FRMenuItemSetClientData](#)  
[FRMenuItemSetComputeEnabledProc](#)  
[FRMenuItemSetComputeMarkedProc](#)  
[FRMenuItemDoExecuteProc](#)

**Note:** If a user-supplied data need to pass to proc, using FRMenuItemSetClientData() to set the user-supplied data first.

**FRMenuItemsetIcon****Syntax**

```
FS_BOOL FRMenuItemsetIcon (
    FR\_MenuItem menuitem,
    FS\_DIBitmap bitmap
);
```

**Description**

Sets the icon of the menu item.

**Parameter**

menuitem	[In] The menu to which the icon is attached.
bitmap	[In] The icon to attach.

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_menuTempl.h: 388

**Related method**

[FRMenuItemGetIcon](#)

## FRMenuItemSetSubMenu

**Syntax**

```
FS_BOOL FRMenuItemSetSubMenu (
    FR\_MenuItem menuitem,
    FR\_Menu submenu
);
```

**Description**

Attaches a sub-menu to a menu item.

**Parameter**

---

menuitem	[In] The menu item for which a submenu is attached.
----------	---

---

submenu	[In] The sub-menu to be attached.
---------	-----------------------------------

---

**Return**

[TRUE](#) if success, otherwise [FALSE](#).

**Head file reference**

fr\_menuTempl.h: 357

**Related method**

[FRMenuItemGetSubMenu](#)

## FRMenuItemSetTitle

**Syntax**

```
FS_BOOL FRMenuItemSetTitle (
    FR\_MenuItem menuitem,
    FS\_LPCWSTR wszTitle
);
```

**Description**

Sets a menu item's title.

**Parameter**

---

menuitem	[In] The menu item whose title is set.
wszTitle	[In] The new menu title. It must be a <a href="#">NULL</a> terminated string.

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_menuTempl.h: 410

**Related method**

[FRMenuItemSetTitle](#)

**FRMenuItemSetToolTip****Syntax**

```
FS_BOOL FRMenuItemSetToolTip (
    FR\_MenuItem menuitem,
    FS\_LPCWSTR wszTip
);
```

**Description**

Sets the tool tip that appears when the mouse over the item area.

**Parameter**


---

menuitem	[In] The menu item whose tooltip is set.
wszTip	[In] The new tool tip. It must be a <a href="#">NULL</a> terminated string.

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_menuTempl.h: 437

**Related method**

[FRMenuItemSetDescribeText](#)

**FRMenuItemSetVisible****Syntax**

```
void FRMenuItemSetVisible (
    FR\_MenuItem menuitem,
    FS\_BOOL bShow
);
```

**Description**

Whether sets the menu item visible or not.

**Parameter**

menuitem	[In] The input menu item.
----------	---------------------------

bShow	[In] Indicates whether sets the menu item visible or not.
-------	---

**Return**

void

**Head file reference**

fr\_menuTempl.h: 604

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FR\_MenuOwnerDrawHandler

**[Return from Used by](#)****Description**

FR MenuOwnerDraw Handler.

**Returned from****[FRMenuRegisterOwnerDrawHandle](#)****Used by****[FRMenuDeleteOwnerDrawHandle](#)****Callbacks****Callbacks summary****[FRMenuItemOnMeasureItem](#)**

It is called by foxit reader to measure the menu item.

**[FRMenuItemOnDrawItem](#)**

It is called by foxit reader when a visual aspect of the menu has changed.

**[FRMenuItemOnInitMenuPopup](#)**

The Foxit Reader calls this member function when a pop-up menu is about to become active. This allows a plug-in to modify the pop-up menu before it is displayed without changing the entire menu.

## Callbacks detail

### FRMenuItemOnMeasureItem

#### Syntax

```
typedef FS_BOOL (*FRMenuItemOnMeasureItem)(  
    FR_MenuItem menuItem,  
    FS_INT32 width,  
    FS_INT32 height  
)
```

#### Description

It is called by foxit reader to measure the menu item.

#### Parameter

menuItem	[In] The menu item.
width	[In] The width of the menu item.
height	[In] The height of the menu item.

#### Return

TRUE means it is handled successfully, otherwise not.

#### Head file reference

fr\_menuExpT.h: 243

#### Group

[FR\\_MenuOwnerDrawCallbacksRec](#)

### FRMenuItemOnDrawItem

#### Syntax

```
typedef FS_BOOL (*FRMenuItemOnDrawItem)(  
    FR_MenuItem menuItem,  
    HDC hDC,  
    const FS_Rect rect  
)
```

#### Description

It is called by foxit reader when a visual aspect of the menu has changed.

#### Parameter



menuItem	[In] The menu item.
hDC	[In] Identifies a device context. This device context must be used when performing drawing operations on the control.
rect	[In] The rectangle of the menu item.

**Return**

[TRUE](#) means it is handled successfully, otherwise not.

**Head file reference**

fr\_menuExpT.h: 257

**Group**

[FR\\_MenuOwnerDrawCallbacksRec](#)

**FRMenuItemOnInitMenuPopup****Syntax**

```
typedef void (*FRMenuItemOnInitMenuPopup)(  
    FR\_Menu menu  
);
```

**Description**

The Foxit Reader calls this member function when a pop-up menu is about to become active. This allows a plug-in to modify the pop-up menu before it is displayed without changing the entire menu.

**Parameter**

menu	[In] The menu to become active.
------	---------------------------------

**Return**

void

**Head file reference**

fr\_menuExpT.h: 269

**Group**

[FR\\_MenuOwnerDrawCallbacksRec](#)

## FR\_MessageBar

[Return from Used by](#)

## Description

The [FR\\_MessageBar](#) object is used to show some customer message. Plug-in can add some element to the ssage bar, such as bitmap, text, button and so on.

### Returned from

[FRMessageBarCreate](#)  
[FRMessageBarGetVisibleMessageBar](#)

### Used by

[FRMessageBarAddButton](#)  
[FRMessageBarAddButton2](#)  
[FRMessageBarAddButtonImage](#)  
[FRMessageBarAddButtonImage2](#)  
[FRMessageBarChangeButton](#)  
[FRMessageBarDestroy](#)  
[FRMessageBarEnableButton](#)  
[FRMessageBarGetButtonImage](#)  
[FRMessageBarIsButtonEnable](#)  
[FRMessageBarIsVisible](#)  
[FRMessageBarSetBitmap](#)  
[FRMessageBarSetBitmap2](#)  
[FRMessageBarSetButtonAlignment](#)  
[FRMessageBarSetButtonDownProc](#)  
[FRMessageBarSetButtonExecuteProc](#)  
[FRMessageBarSetButtonHelpText](#)  
[FRMessageBarSetButtonPressed](#)  
[FRMessageBarSetClientData](#)  
[FRMessageBarSetText](#)  
[FRMessageBarShow](#)

## Enumerations

### Enumerations summary

#### [FRMessageBarElementAlignment](#)

Message bar element alignment type enumeration.

#### [FRMessageBarIconType](#)

Icon type enumeration of the message bar.

### Enumerations detail

#### FRMessageBarElementAlignment

##### Syntax

```
enum FRMessageBarElementAlignment{
    FR_ALIGN_LEFT,
    FR_ALIGN_RIGHT
};
```

##### Description

Message bar element alignment type enumeration.

**Head file reference**

fr\_barExpT.h: 209

**FR\_ALIGN\_LEFT**

Align to left.

**FR\_ALIGN\_RIGHT**

Align to right.

**FRMessageBarIconType****Syntax**

```
enum FRMessageBarIconType{  
    FR\_MessageBar\_ICON\_USER,  
    FR\_MessageBar\_ICON\_WARNING,  
    FR\_MessageBar\_ICON\_OK,  
    FR\_MessageBar\_ICON\_ERROR  
};
```

**Description**

Icon type enumeration of the message bar.

**Head file reference**

fr\_barExpT.h: 218

**FR\_MessageBar\_ICON\_USER**

No icon.

**FR\_MessageBar\_ICON\_WARNING**

Warning icon.

**FR\_MessageBar\_ICON\_OK**

OK icon.

**FR\_MessageBar\_ICON\_ERROR**

Error icon.

## Functions

### Functions summary

**[FRMessageBarAddButton](#)**

Adds a new button to the message bar.

**[FRMessageBarAddButton2](#)**

Adds a new button to the message bar.

**[FRMessageBarAddButtonImage](#)**

Adds a bitmap to the button image list.

**[FRMessageBarAddButtonImage2](#)**

Adds a bitmap to the button image list.

**FRMessageBarChangeButton**

Changes the button information.

**FRMessageBarCountVisibleMessageBars**

Gets the count of visible message bars.

**FRMessageBarCreate**

Creates a new message bar, or NULL if fails. Plug-in can create a message bar in the FRDocOnFrameCreate () callback.

**FRMessageBarDestroy**

Destroys the message bar. Plug-in can destroy a message bar in the FRDocOnFrameDestroy () callback.

**FRMessageBarEnableButton**

If the *bEnable* is TRUE , the button is enabled. Otherwise not.

**FRMessageBarGetButtonImage**

Gets the image index.

**FRMessageBarGetVisibleMessageBar**

Gets the specified visible message bar by index.

**FRMessageBarIsButtonEnable**

Whether the button is enable or not.

**FRMessageBarIsVisible**

Checks whether the message bar is visible or not.

**FRMessageBarSetBitmap**

Sets bitmap to the message bar.

**FRMessageBarSetBitmap2**

Sets bitmap to the message bar.

**FRMessageBarSetButtonAlignment**

Sets the alignment type of the buttons.

**FRMessageBarSetButtonDropDownProc**

Sets the drop-down callback function.

**FRMessageBarSetButtonExecuteProc**

Sets the user-supplied procedure to call to perform the button's intended function.

**FRMessageBarSetButtonHelpText**

Sets the help text to the message bar.

**FRMessageBarSetButtonPressed**

Sets the pressed type to the button.

**FRMessageBarSetClientData**

Sets the user-supplied data for each user-supplied procedure.

**FRMessageBarSetText**

Sets text to the message bar.

**FRMessageBarShow**

Shows the message bar or hides it.

## Functions detail

### FRMessageBarAddButton

#### Syntax

```
void FRMessageBarAddButton (
    FR_MessageBar msgBar,
    FS_LPCSTR lpsName,
    FS_LPCWSTR lpwsText,
    FS_DIBitmap btnBitmap,
    FS_BOOL bHasDropDownArrow,
```

---

```
FS_BOOL bNeedLayout
);
```

**Description**

Adds a new button to the message bar.

**Parameter**

msgBar	[In] The input message bar.
lpsName	[In] The input name of the button.
lpwsText	[In] The input text displayed on the button.
btnBitmap	[In] The input bitmap showed on the button.
bHasDropDownArrow	[In] Whether the button has the dropped down arrow or not.
bNeedLayout	[In] Whether the button need to be laid out.

**Return**

void

**Head file reference**

fr\_barTempl.h: 610

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRMessageBarAddButton2****Syntax**

```
void FRMessageBarAddButton2 (
    FR_MessageBar msgBar,
    FS_LPCSTR lpsName,
    FS_LPCWSTR lpwsText,
    FS_DIBitmap btnBitmap,
    FS_BOOL bHasDropDownArrow,
    FS_BOOL bNeedLayout,
    FS_INT32 cx,
    FS_INT32 cy
);
```

**Description**

Adds a new button to the message bar.

**Parameter**


---

msgBar	[In] The input message bar.
lpsName	[In] The input name of the button.
lpwsText	[In] The input text displayed on the button.
btnBitmap	[In] The input bitmap showed on the button.
bHasDropDownArrow	[In] Whether the button has the dropped down arrow or not.
bNeedLayout	[In] Whether the button need to be laid out.
cx	[In] The width of the bitmap when the DPI is 100%. The default value is -1.
cy	[In] The height of the bitmap when the DPI is 100%. The default value is -1.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 791

**Since**[SDK\\_LATEEST\\_VERSION > 3.0](#)**FRMessageBarAddButtonImage****Syntax**

```
FS_INT32 FRMessageBarAddButtonImage (
    FR\_MessageBar msgBar,
    FS\_DIBitmap pBtnBitmap
);
```

**Description**

Adds a bitmap to the button image list.

**Parameter**

---

msgBar	[In] The input message bar.
--------	-----------------------------

---

pBtnBitmap	[In] Adds a bitmap to the button image list.
------------	--

---

**Return**

The bitmap index in the button image list.

**Head file reference**

fr\_barTempl.h: 710

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRMessageBarAddButtonImage2****Syntax**

```
FS_INT32 FRMessageBarAddButtonImage2 (
    FR\_MessageBar msgBar,
    FS\_DIBitmap pBtnBitmap,
    FS\_INT32 cx,
    FS\_INT32 cy
);
```

**Description**

Adds a bitmap to the button image list.

**Parameter**


---

msgBar	[In] The input message bar.
--------	-----------------------------

---

pBtnBitmap	[In] Adds a bitmap to the button image list.
------------	--

---

cx	[In] The width of the bitmap when the DPI is 100%. The default value is -1.
----	---

---

cy	[In] The height of the bitmap when the DPI is 100%. The default value is -1.
----	--

---

**Return**

The bitmap index in the button image list.

**Head file reference**

fr\_barTempl.h: 808

**Since**

[SDK\\_LATEEST\\_VERSION > 3.0](#)



## FRMessageBarChangeButton

### Syntax

```
void FRMessageBarChangeButton (  
    FR\_MessageBar msgBar,  
    FS\_LPCSTR lpsName,  
    FS\_LPCWSTR lpwsText,  
    FS\_INT32 nImageIndex  
) ;
```

### Description

Changes the button information.

### Parameter

---

msgBar	[In] The input message bar.
--------	-----------------------------

---

lpsName	[In] The input specified button name.
---------	---------------------------------------

---

lpwsText	[In] The text you want to change to.
----------	--------------------------------------

---

nImageIndex	[In] The image index you want to change to.
-------------	---

---

### Return

void.

### Head file reference

[fr\\_barTempl.h: 732](#)

### Since

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRMessageBarCountVisibleMessageBars

### Syntax

```
FS_INT32 FRMessageBarCountVisibleMessageBars (void );
```

### Description

Gets the count of visible message bars.

### Return

The count of visible message bars.

### Head file reference

fr\_barTempl.h: 756

**Since**

[SDK LATEEST VERSION > 2.0](#)

## FRMessageBarCreate

**Syntax**

```
FR_MessageBar FRMessageBarCreate (
    FS_BOOL bShowCloseBtn,
    HWND hFrameWnd,
);
```

**Description**

Creates a new message bar, or [NULL](#) if fails. Plug-in can create a message bar in the [FRDocOnFrameCreate](#) () callback.

**Parameter**

---

bShowCloseBtn	[In] Whether to show the " <i>Close</i> " button on the message bar or not.
hFrameWnd	[In] The input parent frame window.

---

[In]

**Return**

The new [FR\\_MessageBar](#) object.

**Head file reference**

fr\_barTempl.h: 547

## FRMessageBarDestroy

**Syntax**

```
void FRMessageBarDestroy (
    FR_MessageBar msgBar
);
```

**Description**

Destroys the message bar. Plug-in can destroy a message bar in the [FRDocOnFrameDestroy](#) () callback.

**Parameter**

---

msgBar	[In] The input message bar to be destroyed.
--------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 559

**FRMessageBarEnableButton****Syntax**

```
FS_BOOL FRMessageBarEnableButton (
    FR_MessageBar msgBar,
    FS_LPCSTR lpsName,
    FS_BOOL bEnable
);
```

**Description**

If the *bEnable* is [TRUE](#) , the button is enabled. Otherwise not.

**Parameter**

---

msgBar	[In] The input message bar.
--------	-----------------------------

---

lpsName	[In] The button you want to set.
---------	----------------------------------

---

bEnable	[In] Whether the button is enable or not.
---------	---

---

**Return**

[TRUE](#) means success, otherwise failure.

**Head file reference**

fr\_barTempl.h: 673

**Note:**Using FRMessageBarIsButtonEnable() to get it.

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRMessageBarGetButtonImage****Syntax**

```
FS_INT32 FRMessageBarGetButtonImage (
    FR_MessageBar msgBar,
    FS_LPCSTR lpsName
);
```

**Description**

Gets the image index.

**Parameter**

---

msgBar	[In] The input message bar.
--------	-----------------------------

---

lpsName	[In] The input specified button name.
---------	---------------------------------------

---

**Return**

The bitmap index in the button image list.

**Head file reference**

fr\_barTempl.h: 721

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRMessageBarGetVisibleMessageBar****Syntax**

```
FR_MessageBar FRMessageBarGetVisibleMessageBar (
    FS_INT32 nIndex
);
```

**Description**

Gets the specified visible message bar by index.

**Parameter**

---

nIndex	[In] Specifies the index of the visible message bars.
--------	---

---

**Return**

The specified visible message bar.

**Head file reference**

fr\_barTempl.h: 765

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRMessageBarIsButtonEnable****Syntax**

```
FS_BOOL FRMessageBarIsButtonEnable (
    FR_MessageBar msgBar,
    FS_LPCSTR lpsName
);
```

**Description**

Whether the button is enable or not.

**Parameter**

---

msgBar	[In] The input message bar.
--------	-----------------------------

---

lpsName	[In] The input specified button name.
---------	---------------------------------------

---

**Return**

TRUE means button is enable , otherwise not.

**Head file reference**

fr\_barTempl.h: 679

**Note:**Using FRMessageBarEnableButon() to set it.

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRMessageBarIsVisible****Syntax**

```
FS_BOOL FRMessageBarIsVisible (
    FR\_MessageBar msgBar
);
```

**Description**

Checks whether the message bar is visible or not.

**Parameter**

---

msgBar	[In] The input message bar.
--------	-----------------------------

---

**Return**

TRUE means the message bar is visible, otherwise not.

**Head file reference**

fr\_barTempl.h: 578

**FRMessageBarSetBitmap****Syntax**

```
void FRMessageBarSetBitmap (
```



---

```
FR_MessageBar msgBar,
FS_DIBitmap bitmap,
FS_BOOL bStretch,
FRMessageBarElementAlignment bmpAlignment
);
```

**Description**

Sets bitmap to the message bar.

**Parameter**

msgBar	[In] The input message bar.
bitmap	[In] The input bitmap to be set to the message bar.
bStretch	[In] Whether to stretch the bitmap or not.
bmpAlignment	[In] The input bitmap alignment type.

**Return**

void

**Head file reference**

fr\_barTempl.h: 598

**FRMessageBarSetBitmap2****Syntax**

```
void FRMessageBarSetBitmap2 (
    FR_MessageBar msgBar,
    FS_DIBitmap bitmap,
    FS_BOOL bStretch,
    FRMessageBarElementAlignment bmpAlignment,
    FS_BOOL bNeedLayout,
    FS_INT32 cx,
    FS_INT32 cy
);
```

**Description**

Sets bitmap to the message bar.

**Parameter**

msgBar	[In] The input message bar.
--------	-----------------------------

---

bitmap	[In] The input bitmap to be set to the message bar.
bStretch	[In] Whether to stretch the bitmap or not.
bmpAlignment	[In] The input bitmap alignment type.
bNeedLayout	[In] Whether to lay out the message bar when setting bitmap.
cx	[In] The width of the bitmap when the DPI is 100%. The default value is -1.
cy	[In] The height of the bitmap when the DPI is 100%. The default value is -1.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 775

**Since**[SDK LATEEST VERSION > 3.0](#)**FRMessageBarSetButtonAlignment****Syntax**

```
void FRMessageBarSetButtonAlignment (
    FR_MessageBar msgBar,
    FRMessageBarElementAlignment bmpAlignment
);
```

**Description**

Sets the alignment type of the buttons.

**Parameter**


---

msgBar	[In] The input message bar.
bmpAlignment	[In] The input alignment type.

---

**Return**

void

**Head file reference**

---

fr\_barTempl.h: 625

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FRMessageBarSetButtonDropDownProc****Syntax**

```
FS_BOOL FRMessageBarSetButtonDropDownProc (
    FR\_MessageBar msgBar,
    FS\_LPCSTR lpsName,
    FRBtnDropDownProc proc
);
```

**Description**

Sets the drop-down callback function.

**Parameter**


---

msgBar	[In] The input message bar.
--------	-----------------------------

---

lpsName	[In] Specifies the button name.
---------	---------------------------------

---

proc	[In] The input drop-down callback function.
------	---

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_barTempl.h: 636

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FRMessageBarSetButtonExecuteProc****Syntax**

```
FS_BOOL FRMessageBarSetButtonExecuteProc (
    FR\_MessageBar msgBar,
    FS\_LPCSTR lpsName,
    FRExecuteProc proc
);
```

**Description**

Sets the user-supplied procedure to call to perform the button's intended function.

**Parameter**


---

msgBar	[In] The input message bar.
lpsName	[In] The button you want to set.
proc	[In] The user-supplied procedure to call when the button on the message bar is clicked.

---

**Return**

[TRUE](#) means success, otherwise failure.

**Head file reference**

fr\_barTempl.h: 648

**Note:** The user-supplied data to pass to FRExecuteProc() must be set if the callback FRExecuteProc() needs. Using FRMessageBarSetClientData() to set it.

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRMessageBarSetButtonHelpText****Syntax**

```
FS_BOOL FRMessageBarSetButtonHelpText (
    FR\_MessageBar msgBar,
    FS\_LPCSTR lpsName,
    FS\_WideString wsHelpText
);
```

**Description**

Sets the help text to the message bar.

**Parameter**


---

msgBar	[In] The input message bar.
lpsName	[In] The button you want to set.
wsHelpText	[In] The input help text to be set to the message bar.

---

**Return**

[TRUE](#) means success, otherwise failure.

**Head file reference**

fr\_barTempl.h: 661

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRMessageBarSetButtonPressed

**Syntax**

```
FS_BOOL FRMessageBarSetButtonPressed (
    FR_MessageBar msgBar,
    FS_LPCSTR lpsName,
    FS_BOOL bPressed
);
```

**Description**

Sets the pressed type to the button.

**Parameter**

---

msgBar	[In] The input message bar.
--------	-----------------------------

---

lpsName	[In] The button you want to set.
---------	----------------------------------

---

bPressed	[In] Whether the button has the pressed type.
----------	---

---

**Return**

[TRUE](#) means success, otherwise failure.

**Head file reference**

fr\_barTempl.h: 685

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRMessageBarSetClientData

**Syntax**

```
FS_BOOL FRMessageBarSetClientData (
    FR_MessageBar msgBar,
    FS_LPCSTR lpsName,
    void* clientData
);
```

**Description**

Sets the user-supplied data for each user-supplied procedure.

**Parameter**


---

msgBar	[In] The input message bar.
lpsName	[In] The button you want to set.
clientData	[In] A pointer to user-supplied data to pass to <a href="#">FRExecuteProc</a> (), The data type may be a class or a struct that contain each client data to pass to each user-supplied procedure.

---

**Return**

[TRUE](#) means success, otherwise failure.

**Head file reference**

fr\_barTempl.h: 655

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRMessageBarSetText****Syntax**

```
void FRMessageBarSetText (
    FR\_MessageBar msgBar,
    FS\_WideString wsText,
    FRMessageBarElementAlignment textAlignment
);
```

**Description**

Sets text to the message bar.

**Parameter**


---

msgBar	[In] The input message bar.
wsText	[In] The input text to be set to the message bar.
textAlignment	[In] The input text alignment type.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 587

## FRMessageBarShow

### Syntax

```
FS_BOOL FRMessageBarShow (
    FR_MessageBar msgBar,
    FS_BOOL bShow
);
```

### Description

Shows the message bar or hides it.

### Parameter

msgBar	[In] The input message bar.
--------	-----------------------------

bShow	[In] Whether to show the message bar or not.
-------	--

### Return

[TRUE](#) means successful, otherwise not.

### Head file reference

fr\_barTempl.h: 568

## FR\_PageView

### [Return from](#) [Used by](#)

### Description

A [FR\\_PageView](#) is a view of a [FPD\\_Page](#) object. The page view is a part(always a rectangle area) of document view. A [FR\\_PageView](#) must associated with a [FPD\\_Page](#), but a [FPD\\_Page](#) has more than one [FR\\_PageView](#) some times. [FR\\_PageView](#) has methods to deal with annotations, transform coordinates, control screen redrawing.

### Returned from

[FRAannotGetPageView](#)  
[FRDocViewGetCurrentPageView](#)  
[FRDocViewGetPageView](#)  
[FRDocViewGetPageViewAtPoint](#)  
[FRDocViewGetVisiblePageView](#)

### Used by

[FRDocViewIsValidPageView](#)  
[FRTextSelectToolDoLButtonDblClk](#)  
[FRTextSelectToolDoLButtonDown](#)  
[FRTextSelectToolDoLButtonUp](#)  
[FRTextSelectToolDoMouseMove](#)

[\*\*FRTextSelectToolDoMouseWheel\*\*](#)  
[\*\*FRTextSelectToolDoRButtonUp\*\*](#)  
[\*\*FRPageViewAddAnnot\*\*](#)  
[\*\*FRPageViewCountAnnot\*\*](#)  
[\*\*FRPageViewDeleteAnnot\*\*](#)  
[\*\*FRPageViewDevicePointToPage\*\*](#)  
[\*\*FRPageViewDeviceRectToPage\*\*](#)  
[\*\*FRPageViewDidContentChanged\*\*](#)  
[\*\*FRPageViewDidContentChanged2\*\*](#)  
[\*\*FRPageViewDidContentChanged3\*\*](#)  
[\*\*FRPageViewDidTextObjectChanged\*\*](#)  
[\*\*FRPageViewGetAnnotAtPoint\*\*](#)  
[\*\*FRPageViewGetAnnotByIndex\*\*](#)  
[\*\*FRPageViewGetCurrentMatrix\*\*](#)  
[\*\*FRPageViewGetDocument\*\*](#)  
[\*\*FRPageViewGetDocView\*\*](#)  
[\*\*FRPageViewGetFocusAnnot\*\*](#)  
[\*\*FRPageViewGetHWnd\*\*](#)  
[\*\*FRPageViewGetPageIndex\*\*](#)  
[\*\*FRPageViewGetPageRect\*\*](#)  
[\*\*FRPageViewGetPageScale\*\*](#)  
[\*\*FRPageViewGetPageState\*\*](#)  
[\*\*FRPageViewGetPageVisibleRect\*\*](#)  
[\*\*FRPageViewGetPDPPage\*\*](#)  
[\*\*FRPageViewGetRenderOptions\*\*](#)  
[\*\*FRPageViewIsVisible\*\*](#)  
[\*\*FRPageViewPointToDevice\*\*](#)  
[\*\*FRPageViewRectToDevice\*\*](#)  
[\*\*FRPageViewUpdateAllViews\*\*](#)

## Enumerations

### Enumerations summary

#### [\*\*FR\\_ContentChangeType\*\*](#)

The content change type. See [FRPageViewDidContentChanged3](#) .

### Enumerations detail

#### FR\_ContentChangeType

##### Syntax

```
enum FR_ContentChangeType{
    FR\_ContentChangeType\_None,
    FR\_ContentChangeType\_Add,
    FR\_ContentChangeType\_Delete,
    FR\_ContentChangeType\_Modify
};
```

##### Description

The content change type. See [FRPageViewDidContentChanged3](#) .

##### Head file reference

fr\_viewExpT.h: 228

**FR\_ContentChangeType\_None**

**FR\_ContentChangeType\_Add**

**FR\_ContentChangeType\_Delete**

**FR\_ContentChangeType\_Modify**

## Structures

### Structures summary

#### [FR\\_ChangedContentRec](#)

A data structure to store the content changed in the page view. See [FRPageViewDidContentChanged3](#).

#### [FR\\_WinPortRec](#)

A data structure for *Windows*. See [FRDocViewGetMachinePort](#). See [FRDocViewReleaseMachinePort](#).

### Structs detail

#### FR\_ChangedContentRec

##### Syntax

```
typedef struct _fr_changedcontent_{
    void* pDocView,
    FS_INT32 nIndex,
    void* pPagepos,
    FS_BOOL bInFormXObject,
    FS_DWordArray arrayFormLayer
}FR_ChangedContentRec, *FR_ChangedContent;
```

##### Description

A data structure to store the content changed in the page view. See [FRPageViewDidContentChanged3](#).

##### Head file reference

fr\_viewExpT.h: 241

##### pDocView

A data structure to store the content changed in the page view. See [FRPageViewDidContentChanged3](#).

##### nPageIndex

A data structure to store the content changed in the page view. See [FRPageViewDidContentChanged3](#).

##### pPagepos

A data structure to store the content changed in the page view. See [FRPageViewDidContentChanged3](#).

##### bInFormXObject

A data structure to store the content changed in the page view. See [FRPageViewDidContentChanged3](#).

#### arrayFormLayer

A data structure to store the content changed in the page view. See [FRPageViewDidContentChanged3](#).

## FR\_WinPortRec

#### Syntax

```
typedef struct _fr_winport_{
    HWND hWnd,
    HDC hDC
}FR_WinPortRec, *FR_WinPort;
```

#### Description

A data structure for *Windows*. See [FRDocViewGetMachinePort](#). See [FRDocViewReleaseMachinePort](#).

#### Head file reference

fr\_viewExpT.h: 215

#### hWnd

Handle to a window.

#### hDC

Handle to a device context.

## Functions

### Functions summary

#### [FRPageViewAddAnnot](#)

Adds the annotation to the page.

#### [FRPageViewCountAnnot](#)

Gets the number of annotations associated with specified page view. This interface is not available in version 1.0.

#### [FRPageViewDeleteAnnot](#)

Deletes the specified annotation.

#### [FRPageViewDevicePointToPage](#)

Transforms a point's coordinate from device space to user space.

#### [FRPageViewDeviceRectToPage](#)

Transforms a rectangle from device space to user space.

#### [FRPageViewDidContentChanged](#)

The content of [FPD\\_Page](#) of the *pv* has been modified. Foxit Reader will broadcast some notifications to plug-ins.

#### [FRPageViewDidContentChanged2](#)

The content of [FPD\\_Page](#) of the *pv* has been modified. Foxit Reader will broadcast some notifications to plug-ins.

#### [FRPageViewDidContentChanged3](#)

The content of [FPD\\_Page](#) of the *pv* has been modified. Foxit Reader will broadcast some notifications to plug-ins.

**[FRPageViewDidTextObjectChanged](#)**

The text objects of [FPD\\_Page](#) of the *pv* has been modified. Foxit Reader will broadcast some notifications to plug-ins.

**[FRPageViewGetAnnotAtPoint](#)**

Gets the annotation at the specified point.

**[FRPageViewGetAnnotByIndex](#)**

Gets the annotation by index.

**[FRPageViewGetCurrentMatrix](#)**

Gets the current matrix of page view.

**[FRPageViewGetDocument](#)**

Gets the [FR\\_Document](#) for the document currently displayed in specified pageview.

**[FRPageViewGetDocView](#)**

Gets the [FR\\_DocView](#) for specified pageview.

**[FRPageViewGetFocusAnnot](#)**

Gets the focus annotation.

**[FRPageViewGetHWnd](#)**

Gets the HWND handler in which a page view is displaying.

**[FRPageViewGetPageIndex](#)**

Gets the current page number.

**[FRPageViewGetPageRect](#)**

Gets the current bounding-box of specified page view.

**[FRPageViewGetPageScale](#)**

Gets the page scale.

**[FRPageViewGetPageState](#)**

Gets the page state. Sets *outDestArray* NULL to get the count of the destination array first.

**[FRPageViewGetPageVisibleRect](#)**

Gets the current visible region of specified page view.

**[FRPageViewGetPDPAGE](#)**

Gets a [FPD\\_Page](#) currently displayed in the specified page view. This do not new a [FPD\\_Page](#) object. Do not use this result across methods that might change the current page. To obtain a value that can be used across such calls, use [FPD\\_PageNew](#) instead.

**[FRPageViewGetRenderOptions](#)**

Gets the rendering options.

**[FRPageViewIsVisible](#)**

Gets a flag that indicate whether a page view is visible.

**[FRPageViewPointToDevice](#)**

Transforms a point from user space to device space.

**[FRPageViewRectToDevice](#)**

Transforms a rectangle from user space to device space.

**[FRPageViewUpdateAllViews](#)**

Updates all the page views where the annotation is shown.

## Functions detail

### [FRPageViewAddAnnot](#)

#### Syntax

```
FR_Annot FRPageViewAddAnnot (
    FR_PageView pv,
    FPD_Object annotDict,
```

---

```
FS_INT32 nIndex
);
```

**Description**

Adds the annotation to the page.

**Parameter**


---

pv	[In] The input page view object.
annotDict	[In] The input annotation dictionary you want to add.
nIndex	[In] The index where you want to add the annotation. Sets it -1 as default.

---

**Return**

The UI layer annotation object.

**Head file reference**

fr\_viewTempl.h: 733

**Related method****Since**

[SDK LATEEST VERSION > 7.3.1](#)

**FRPageViewCountAnnot****Syntax**

```
FS_INT32 FRPageViewCountAnnot (
    FR_PageView pv
);
```

**Description**

Gets the number of annotations associated with specified page view. This interface is not available in version 1.0.

**Parameter**


---

pv	[In] The page view whose annotation count is obtained.
----	--

---

**Return**

The number of annotations associated with *pv*.

**Head file reference**

fr\_viewTempl.h: 510

## FRPageViewDeleteAnnot

### Syntax

```
FS_BOOL FRPageViewDeleteAnnot (
    FR\_PageView pv,
    FR\_Annot frAnnot
);
```

### Description

Deletes the specified annotation.

### Parameter

---

pv	[In] The input page view object.
----	----------------------------------

---

frAnnot	[In] The input annotation to be deleted.
---------	--

---

### Return

TRUE for success, otherwise failure.

### Head file reference

fr\_viewTempl.h: 674

### Related method

#### Since

[SDK LATEEST VERSION > 2.1.0.4](#)

## FRPageViewDevicePointToPage

### Syntax

```
void FRPageViewDevicePointToPage (
    FR\_PageView pv,
    FS\_INT32 window_x,
    FS\_INT32 window_y,
    FS\_FloatPoint* outPt
);
```

### Description

Transforms a point's coordinate from device space to user space.

### Parameter

---

pv	[In] The page view for which the point's coordinates are transformed from device space to user space.
----	---

---

---

window_x	[In] The x-coordinate of the point to transform, specified in device space coordinates.
----------	---

---

window_y	[In] The y-coordinate of the point to transform, specified in device space coordinates.
----------	---

---

outPt	[Out] (Filled by this method) A pointer to a point whose user space coordinates corresponding <i>window_x</i> and <i>window_y</i> .
-------	---

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 528

**Related method**[FRPageViewDeviceRectToPage](#)[FRPageViewPointToDevice](#)[FRPageViewRectToDevice](#)**FRPageViewDeviceRectToPage****Syntax**

```
void FRPageViewDeviceRectToPage (
    FR\_PageView pv,
    const FS\_Rect* rect,
    FS\_FloatRect* outRect
);
```

**Description**

Transforms a rectangle from device space to user space.

**Parameter**


---

pv	[In] The page view for which the rectangle is transformed from device space to user space.
----	--

---

rect	[In] A pointer to device space rectangle whose coordinates are transformed to user space.
------	---

---

outRect	[Out] (Filled by this method) A pointer to a user space rectangle corresponding <i>rect</i> .
---------	---

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 537

**Related method**[DevicePointToPage](#)[FRPageViewPointToDevice](#)[FRPageViewRectToDevice](#)**FRPageViewDidContentChanged****Syntax**

```
void FRPageViewDidContentChanged (
    FR\_PageView pv,
    FS\_BOOL bReloadPage
);
```

**Description**

The content of [FPD\\_Page](#) of the *pv* has been modified. Foxit Reader will broadcast some notifications to plug-ins.

**Parameter**

<i>pv</i>	[In] The page view whose <a href="#">FPD_Page</a> has been modified.
<i>bReloadPage</i>	[In] A flag to force reader to reload the page data. If <a href="#">TRUE</a> , Foxit Reader will reload the page data and reparse the page. The flag must be <a href="#">TRUE</a> and call <a href="#">FPDPageGenerateContent()</a> before calling this method if the <a href="#">FPD_Page</a> whose content is modified is not using <a href="#">FRPageViewGetPDPPage()</a> to obtain.

**Return**

void

**Head file reference**

fr\_viewTempl.h: 616

**Related method****FRPageViewDidContentChanged2****Syntax**

```
void FRPageViewDidContentChanged2 (
    FR\_PageView pv,
    FS\_BOOL bReloadPage,
    FS\_BOOL bResizePageNotify,
    void* pChangeData
);
```

**Description**

The content of [FPD\\_Page](#) of the *pv* has been modified. Foxit Reader will broadcast some notifications to plug-ins.

**Parameter**


---

pv	[In] The page view whose <a href="#">FPD_Page</a> has been modified.
bReLoadPage	[In] A flag to force reader to reload the page data. If <a href="#">TRUE</a> , Foxit Reader will reload the page data and reparse the page. The flag must be <a href="#">TRUE</a> and call <a href="#">FPDPageGenerateContent()</a> before calling this method if the <a href="#">FPD_Page</a> whose content is modified is not using <a href="#">FRPageViewGetPDPPage()</a> to obtain.
bResizePageNotify	[In] Checks whether to broadcast the notification of page resizing.
pChangeData	[In] It indicates what data is changed.

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 757

**Related method****Since**

[SDK LATEEST VERSION > 8.3.2](#)

**FRPageViewDidContentChanged3****Syntax**

```
void FRPageViewDidContentChanged3 (
    FR\_PageView pv,
    FS\_BOOL bReLoadPage,
    FS\_BOOL bResizePageNotify,
    FS\_PtrArray objArray,
    FR\_ContentChangeType changeType
);
```

**Description**

The content of [FPD\\_Page](#) of the *pv* has been modified. Foxit Reader will broadcast some notifications to plug-ins.

**Parameter**

---

pv	[In] The page view whose <a href="#">FPD_Page</a> has been modified.
bReLoadPage	[In] A flag to force reader to reload the page data. If <a href="#">TRUE</a> , Foxit Reader will reload the page data and reparse the page. The flag must be <a href="#">TRUE</a> and call <a href="#">FPDPageGenerateContent()</a> before calling this method if the <a href="#">FPD_Page</a> whose content is modified is not using <a href="#">FRPageViewGetPDPPage()</a> to obtain.
bResizePageNotify	[In] Checks whether to broadcast the notification of page resizing.
objArray	[In] It indicates what data is changed. The value type in the array is <a href="#">FR_ChangedContent</a> .
changeType	[In] The content change type.

---

**Return**  
void**Head file reference**  
fr\_viewTempl.h: 772**Related method****Since**  
[SDK\\_LATEEST\\_VERSION > 9.1](#)**FRPageViewDidTextObjectChanged****Syntax**

```
void FRPageViewDidTextObjectChanged (
    FR\_PageView pv
);
```

**Description**

The text objects of [FPD\\_Page](#) of the *pv* has been modified. Foxit Reader will broadcast some notifications to plug-ins.

**Parameter**


---

pv	[In] The page view whose text objects of <a href="#">FPD_Page</a> has been modified.
----	--

---

**Return**  
void**Head file reference**

fr\_viewTempl.h: 638

### Related method

#### FRPageViewGetAnnotAtPoint

##### Syntax

```
FR_Annot FRPageViewGetAnnotAtPoint (
    FR\_PageView pv,
    FS\_DevicePoint point,
    FS\_LPCSTR pszSubType
);
```

##### Description

Gets the annotation at the specified point.

##### Parameter

pv	[In] The input page view object.
----	----------------------------------

point	[In] The input point where to get the annotation.
-------	---

pszSubType	[In] Specifies the sub type of the annotation you want to get. You can set it NULL as default.
------------	---

##### Return

The specified annotation.

##### Head file reference

fr\_viewTempl.h: 708

### Related method

##### Since

[SDK LATEEST VERSION > 7.3](#)

#### FRPageViewGetAnnotByIndex

##### Syntax

```
FR_Annot FRPageViewGetAnnotByIndex (
    FR\_PageView pv,
    FS\_INT32 index
);
```

##### Description

Gets the annotation by index.

**Parameter**

---

pv	[In] The input page view object.
----	----------------------------------

---

index	[In] The specified index of the annotations.
-------	--

---

**Return**

The specified annotation.

**Head file reference**

fr\_viewTempl.h: 648

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRPageViewGetCurrentMatrix****Syntax**

```
FS_AffineMatrix FRPageViewGetCurrentMatrix (
    FR\_PageView pv
);
```

**Description**

Gets the current matrix of page view.

**Parameter**

---

pv	[In] The page view whose matrix is obtained.
----	--

---

**Return**

Current matrix of *pv*.

**Head file reference**

fr\_viewTempl.h: 519

**FRPageViewGetDocument****Syntax**

```
FR_Document FRPageViewGetDocument (
    FR\_PageView pv
);
```

**Description**

Gets the [FR\\_Document](#) for the document currently displayed in specified pageview.

**Parameter**

---

pv	[In] The page view whose <a href="#">FR_Document</a> is obtained.
----	---

---

**Return**

The [FR\\_Document](#) for *pv*.

**Head file reference**

fr\_viewTempl.h: 29

**Related method**

[FRDocViewGetDocument](#)

[FRPageViewGetDocView](#)

**FRPageViewGetDocView****Syntax**

```
FR_DocView FRPageViewGetDocView (
    FR\_PageView pv
);
```

**Description**

Gets the [FR\\_DocView](#) for specified pageview.

**Parameter**

---

pv	[In] The page view whose <a href="#">FR_DocView</a> is obtained.
----	--

---

**Return**

The [FR\\_DocView](#) for *pv*.

**Head file reference**

fr\_viewTempl.h: 470

**Related method**

[FRDocGetDocView](#)

[FRDocGetCurrentDocView](#)

[FRDocViewGetPageView](#)

**FRPageViewGetFocusAnnot****Syntax**

```
FR_Annot FRPageViewGetFocusAnnot (
    FR\_PageView pv
);
```

**Description**

Gets the focus annotation.

**Parameter**

---

pv	[In] The input page view object.
----	----------------------------------

---

**Return**

The focus annotation.

**Head file reference**

fr\_viewTempl.h: 686

**Related method****Since**

[SDK LATEEST VERSION > 3.0.0.0](#)

**FRPageViewGetHWnd****Syntax**

```
HWND FRPageViewGetHWnd (
    FR\_PageView pv
);
```

**Description**

Gets the HWND handler in which a page view is displaying.

**Parameter**

---

pv	[In] The page view which HWND will be obtained.
----	---

---

**Return**

The HWND handler of *pv*.

**Head file reference**

fr\_viewTempl.h: 628

**Related method****FRPageViewGetPageIndex****Syntax**

```
FS_INT32 FRPageViewGetPageIndex (
    FR\_PageView pv
);
```

**Description**

Gets the current page number.

**Parameter**

---

pv	[In] The page view whose current page number is obtained.
----	---

---

**Return**

The current page number, or [NULL](#) if *pv* is invalid. The first page number of a document is page 0.

**Head file reference**

fr\_viewTempl.h: 500

**FRPageViewGetPageRect****Syntax**

```
FS_Rect FRPageViewGetPageRect (
    FR\_PageView pv
);
```

**Description**

Gets the current bounding-box of specified page view.

**Parameter**

---

pv	[In] The page view whose bounding-box to obtained.
----	--

---

**Return**

The current bounding-box for *pv*.

**Head file reference**

fr\_viewTempl.h: 596

**Related method**

[FRPageViewGetPageVisibleRect](#)

**Note:** The page bounding-box specified in device space coordinates.

**FRPageViewGetPageScale****Syntax**

```
FS_FLOAT FRPageViewGetPageScale (
    FR\_PageView pv
);
```

**Description**

Gets the page scale.

**Parameter**

---

pv	[In] The input page view object.
----	----------------------------------

---

**Return**

The page scale.

**Head file reference**

fr\_viewTempl.h: 697

**Related method****Since**

[SDK LATEEST VERSION > 3.0.0.0](#)

**FRPageViewGetPageState****Syntax**

```
void FRPageViewGetPageState (
    FR\_PageView pv,
    FS\_INT32* outFitType,
    FS\_FLOAT* outDestArray,
    FS\_INT32* outDestArrayCount
);
```

**Description**

Gets the page state. Sets *outDestArray* NULL to get the count of the destination array first.

**Parameter**

---

pv	[In] The input page view object.
----	----------------------------------

---

outFitType	[Out] It receives the fit type.
------------	---------------------------------

---

outDestArray	[Out] It receives the destination array.
--------------	--

---

outDestArrayCount	[Out] It receives the count of the destination array.
-------------------	---

---

**Return**

void

**Head file reference**

---

fr\_viewTempl.h: 660

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

### FRPageViewGetPageVisibleRect

#### Syntax

```
FS_FloatRect FRPageViewGetPageVisibleRect (
    FR\_PageView pv
);
```

#### Description

Gets the current visible region of specified page view.

#### Parameter

pv	[In] The page view whose visible rectangle to obtained.
----	---

#### Return

A rectangle specified in user space, instead of the current visible region of *pv*.

#### Head file reference

fr\_viewTempl.h: 601

#### Related method

[FRPageViewGetPageRect](#)

**Note:** If the specified page view is invisible, the returned rectangle is empty.

### FRPageViewGetPDPage

#### Syntax

```
FPD_Page FRPageViewGetPDPage (
    FR\_PageView pv
);
```

#### Description

Gets a [FPD Page](#) currently displayed in the specified page view. This do not new a [FPD Page](#) object. Do not use this result across methods that might change the current page. To obtain a value that can be used across such calls, use [FPD PageNew](#) instead.

#### Parameter

---

pv	[In] The page view whose <a href="#">FPD_Page</a> is obtained.
----	--

---

**Return**

[FPD\\_Page](#) displayed in *pv* , or [NULL](#) if no valid [FPD\\_Page](#) associated with *pv* .

**Head file reference**

fr\_viewTempl.h: 487

**Related method**

[FRDocGetPDDoc](#)

**FRPageViewGetRenderOptions****Syntax**

```
FPD_RenderOptions FRPageViewGetRenderOptions (
    FR\_PageView pv
);
```

**Description**

Gets the rendering options.

**Parameter**

---

pv	[In] The input page view object.
----	----------------------------------

---

**Return**

The rendering options.

**Head file reference**

fr\_viewTempl.h: 746

**Related method****Since**

[SDK LATEEST VERSION > 8.2.1](#)

**FRPageViewIsVisible****Syntax**

```
FS_BOOL FRPageViewIsVisible (
    FR\_PageView pv
);
```

**Description**

Gets a flag that indicate whether a page view is visible.

**Parameter**

---

pv	[In] The page view to adjust whether it is visible.
----	---

---

**Return**

[TRUE](#) if *pv* is visible, otherwise [FALSE](#).

**Head file reference**

fr\_viewTempl.h: 587

**FRPageViewPointToDevice****Syntax**

```
void FRPageViewPointToDevice (
    FR\_PageView pv,
    const FS\_FloatPoint* pt,
    FS\_INT32* out_window_x,
    FS\_INT32* out_window_y
);
```

**Description**

Transforms a point from user space to device space.

**Parameter**


---

pv	[In] The page view for which the point's coordinates are transformed from user space to device space.
----	---

---

pt	[In] A pointer to a <a href="#">FS_FloatPoint</a> whose coordinates, specified in device space coordinates, are transformed.
----	--

---

out_window_x	[Out] (Filled by the method) The x-coordinate of the device space point corresponding to <i>pt</i> .
--------------	--

---

out_window_y	[Out] (Filled by the method) The y-coordinate of the device space point corresponding to <i>pt</i> .
--------------	--

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 538

**Related method**

[DevicePointToPage](#)

[DeviceRectToPage](#)

[FRPageViewRectToDevice](#)

**FRPageViewRectToDevice****Syntax**

```
void FRPageViewRectToDevice (
    FR\_PageView pv,
    const FS\_FloatRect* rect,
    FS\_Rect* outRect
);
```

**Description**

Transforms a rectangle from user space to device space.

**Parameter**


---

pv	[In] The page view for which the rectangle is transformed from user space to device space.
rect	[In] A pointer to user space rectangle whose coordinates are transformed to device space.
outRect	[Out] (Filled by this method) A pointer to a device space rectangle corresponding <i>rect</i> .

---

**Return**

void

**Head file reference**

[fr\\_viewTempl.h](#): 539

**Related method**

[DevicePointToPage](#)

[DeviceRectToPage](#)

[FRPageViewPointToDevice](#)

**FRPageViewUpdateAllViews****Syntax**

```
void FRPageViewUpdateAllViews (
    FR\_PageView pv,
    FR\_Annot frAnnot
);
```

**Description**

Updates all the page views where the annotation is shown.

**Parameter**

---

pv	[In] The input page view object.
----	----------------------------------

---

frAnnot	[In] The input annotation you want to update.
---------	---

---

**Return**

void.

**Head file reference**

fr\_viewTempl.h: 721

**Related method****Since**[SDK LATEEST VERSION > 7.3](#)

## FR\_PanelMgr

**[Return from Used by](#)****Description**

The [FR\\_PanelMgr](#) object is used to manage the navigation panels. You can get an existing panel manager through [FRPanelMgrGetPanelMgrFromChildFrm](#) . And can also create a new one through [FRPanelMgrNewPanelMgr](#) .

**Returned from**

[FRPanelMgrGetPanelMgrFromChildFrm](#)  
[FRPanelMgrNewPanelMgr](#)

**Used by**

[FRPanelMgrAddPanel](#)  
[FRPanelMgrAddPanel2](#)  
[FRPanelMgrCreate](#)  
[FRPanelMgrDockToFrameWindow](#)  
[FRPanelMgrGetEnabledAlignment](#)  
[FRPanelMgrGetPanelCount](#)  
[FRPanelMgrGetPanelHwndByName](#)  
[FRPanelMgrGetPanelNameByIndex](#)  
[FRPanelMgrGetPanelTabNameByPt](#)  
[FRPanelMgrGetPanelTabRectByName](#)  
[FRPanelMgrGetPanelTabRectByPt](#)  
[FRPanelMgrGetPanelViewByName](#)  
[FRPanelMgrGetParentFrame](#)  
[FRPanelMgrHasHistory](#)  
[FRPanelMgrHasPanelFloating](#)  
[FRPanelMgrHidePanelTabByName](#)  
[FRPanelMgrIsAllPanelHide](#)  
[FRPanelMgrIsPanelFloating](#)

[FRPanelMgrIsPanelHide](#)  
[FRPanelMgrIsPanelSpreadOut](#)  
[FRPanelMgrLockAllPanel](#)  
[FRPanelMgrRedockAllFloatToInitial](#)  
[FRPanelMgrResetAllPanels](#)  
[FRPanelMgrShowAllPanel](#)  
[FRPanelMgrShowPanelByName](#)  
[FRPanelMgrShowPanelByName2](#)  
[FRPanelMgrShrinkPanelByName](#)

## Definitions

### Definitions summary

#### [FR\\_PANEL\\_LOCATION\\_LEFT](#)

The panel location is left.

#### [FR\\_PANEL\\_LOCATION\\_TOP](#)

The panel location is top.

### Definitions detail

#### **FR\_PANEL\_LOCATION\_LEFT**

##### **Syntax**

```
#define FR_PANEL_LOCATION_LEFT 2
```

##### **Description**

The panel location is left.

##### **Group**

[FRPanelLocation](#)

##### **Head file reference**

fr\_appExpT.h: 6372

#### **FR\_PANEL\_LOCATION\_TOP**

##### **Syntax**

```
#define FR_PANEL_LOCATION_TOP 1
```

##### **Description**

The panel location is top.

##### **Group**

[FRPanelLocation](#)

##### **Head file reference**

fr\_appExpT.h: 6370

## Callbacks

### Callbacks summary

#### [FRPanelEventOnShowPanelMenu](#)

A callback for navigation panel event handler.

#### [FRPanelEventOnPanelActive](#)

A callback for navigation panel event handler.

#### [FRPanelEventOnPanelSize](#)

A callback for navigation panel event handler.

#### [FRPanelEventOnMouseMove](#)

A callback for navigation panel event handler.

### Callbacks detail

#### [FRPanelEventOnShowPanelMenu](#)

##### Syntax

```
typedef FS_BOOL (*FRPanelEventOnShowPanelMenu)(  
    FS_LPVVOID clientData,  
    HWND hParent,  
    FS_INT32 posX,  
    FS_INT32 posY  
)
```

##### Description

A callback for navigation panel event handler. It is called when a context menu is shown on the navigation panel.

##### Parameter

clientData	[In] The user-supplied data.
------------	------------------------------

hParent	[In] The parent window of the navigation panel.
---------	---

posX	[In] The horizon position of the context menu.
------	--

posY	[In] The vertical position of the context menu.
------	---

##### Return

TRUE for success, otherwise failure.

##### Head file reference

fr\_appExpT.h: 6404

##### Group

[FR\\_PanelEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRPanelEventOnPanelActive****Syntax**

```
typedef void (*FRPanelEventOnPanelActive)(  
    FS\_LPVOID clientData,  
    FS\_LPCSTR lpsName  
)
```

**Description**

A callback for navigation panel event handler. It is called when a navigation panel is activated.

**Parameter**

clientData	[In] The user-supplied data.
lpsName	[In] The name of the navigation panel.

**Return**

void.

**Head file reference**

fr\_appExpT.h: 6419

**Group**

[FR\\_PanelEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRPanelEventOnPanelSize****Syntax**

```
typedef void (*FRPanelEventOnPanelSize)(  
    FS\_LPVOID clientData,  
    FS\_LPCSTR lpsName,  
    FS\_Rect rect,  
    HWND hChildFrm  
)
```

**Description**

A callback for navigation panel event handler. It is called when the size of the navigation panel is changing.

**Parameter**


---

clientData	[In] The user-supplied data.
lpsName	[In] The name of the navigation panel.
rect	[In] The size of the navigation panel.
hChildFrm	[In] The child frame the navigation panel attached to.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 6436

**Group**[FR\\_PanelEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRPanelEventOnMouseMove****Syntax**

```
typedef void (*FRPanelEventOnMouseMove)(  
    FS\_LPVVOID clientData,  
    FS\_INT32 x,  
    FS\_INT32 y  
)
```

**Description**

A callback for navigation panel event handler. It is called when the mouse is moving on the navigation panel.

**Parameter**


---

clientData	[In] The user-supplied data.
x	[In] The horizon position of the mouse.
y	[In] The vertical position of the mouse.

---

**Return**

void.

**Head file reference**

fr\_appExpT.h: 6452

**Group**[FR\\_PanelEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)

## Functions

### Functions summary

[FRPanelMgrAddPanel](#)

Adds a panel.

[FRPanelMgrAddPanel2](#)

Adds a panel.

[FRPanelMgrCreate](#)Creates the panel manager after [FRPanelMgrNewPanelMgr](#) .[FRPanelMgrCreatePanelEventHandler](#)

Creates the pointer to the panel event handler.

[FRPanelMgrDeletePanelMgr](#)

Deletes the panel manager.

[FRPanelMgrDestroyPanelEventHandler](#)

Destroys the panel event handler.

[FRPanelMgrDockToFrameWindow](#)

Docks the panel to the frame window.

[FRPanelMgrGetEnabledAlignment](#)

Gets the enabled alignment. References to MFC ControlBar styles.

[FRPanelMgrGetPanelCount](#)

Gets the panel count.

[FRPanelMgrGetPanelHwndByName](#)

Gets the window handle to the panel by name.

[FRPanelMgrGetPanelMgrFromChildFrm](#)

Gets the specified panel manager.

[FRPanelMgrGetPanelNameByIndex](#)

Gets the specified panel name.

[FRPanelMgrGetPanelTabNameByPt](#)

Gets the panel name by point.

[FRPanelMgrGetPanelTabRectByName](#)

Gets the rectangle of the panel tab by name.

[FRPanelMgrGetPanelTabRectByPt](#)

Gets the rectangle of the panel tab by point.

[FRPanelMgrGetPanelViewByName](#)

Gets the specified panel view handle.

[FRPanelMgrGetParentFrame](#)

Gets the parent window handle.

[FRPanelMgrHasHistory](#)

Checks whether the customer has configured the panel or not.

**FRPanelMgrHasPanelFloating**

Checks whether any panel is floating or not.

**FRPanelMgrHidePanelTabByName**

Hides the specified panel by name.

**FRPanelMgrIsAllPanelHide**

Checks whether all the panels are hidden or not.

**FRPanelMgrIsPanelFloating**

Checks whether the panel is floating or not.

**FRPanelMgrIsPanelHide**

Checks whether the specified panel is hidden or not.

**FRPanelMgrIsPanelSpreadOut**

Checks whether the panel is spread out or not.

**FRPanelMgrLockAllPanel**

Locks all the panels.

**FRPanelMgrNewPanelMgr**

Creates an instance of the panel manager. You have to invoke [FRPanelMgrCreate](#) then.

**FRPanelMgrRedockAllFloatToInitial**

All floating panel is restored to the left side.

**FRPanelMgrResetAllPanels**

Resets all the panels.

**FRPanelMgrShowAllPanel**

Shows all the panels or not.

**FRPanelMgrShowPanelByName**

Shows the specified panel or not.

**FRPanelMgrShowPanelByName2**

Shows the specified panel or not.

**FRPanelMgrShrinkPanelByName**

Shrink the specified panel by name.

## Functions detail

### FRPanelMgrAddPanel

#### Syntax

```
HWND FRPanelMgrAddPanel (
    FR_PanelMgr panelMgr,
    FS_LPCWSTR csToolTip,
    FS_LPCWSTR csText,
    FS_LPCSTR csName,
    FS_DIBitmap pBitmap,
    FS_BOOL bInitialShow,
    FS_BOOL bDockBottom
);
```

#### Description

Adds a panel.

#### Parameter

---

panelMgr	[In] The input panel manager.
csToolTip	[In] The tool tip of the panel.
csText	[In] The display text of the panel.
csName	[In] The name of the panel.
pBitmap	[In] The icon of the panel.
bInitialShow	[In] Whether to show the panel at the first time.
bDockBottom	[In] Whether the panel is docked to the bottom or not.

---

**Return**

The handle to the panel.

**Head file reference**

fr\_appTempl.h: 3863

**Related method****Since**

[SDK LATEEST VERSION > 7.1](#)

**FRPanelMgrAddPanel2****Syntax**

```
HWND FRPanelMgrAddPanel2 (
    FR\_PanelMgr panelMgr,
    FS\_LPCWSTR csToolTip,
    FS\_LPCWSTR csText,
    FS\_LPCSTR csName,
    FS\_DIBitmap pBitmap,
    FS\_BOOL bInitialShow,
    FS\_BOOL bDockBottom,
    FS\_INT32 nPos
);
```

**Description**

Adds a panel.

**Parameter**


---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

---

csToolTip	[In] The tool tip of the panel.
csText	[In] The display text of the panel.
csName	[In] The name of the panel.
pBitmap	[In] The icon of the panel.
bInitialShow	[In] Whether to show the panel at the first time.
bDockBottom	[In] Whether the panel is docked to the bottom or not.
nPos	[In] The position where the panel will be added.

---

**Return**

The handle to the panel.

**Head file reference**

fr\_appTempl.h: 4119

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 9.0.1](#)

**FRPanelMgrCreate****Syntax**

```
FS_BOOL FRPanelMgrCreate (
    FR\_PanelMgr panelMgr,
    void* pParentWnd,
    FS_INT32 nTablocation,
    FS_DWORD dwEnabledAlignment,
    void* panelEventHandler
);
```

**Description**

Creates the panel manager after [FRPanelMgrNewPanelMgr](#) .

**Parameter**


---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

---

pParentWnd	[In] The input parent window.
nTablocation	[In] Specifies the tab location.
dwEnabledAlignment	[In] Specifies the enabled alignment. References to MFC ControlBar styles.
panelEventHandler	[In] The input panel event handler to receive the event occurred.

---

**Return**

True for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 3793

**Related method**

[FRPanelMgrNewPanelMgr](#)

**Since**

[SDK LATEEST VERSION > 7.1](#)

**FRPanelMgrCreatePanelEventHandler****Syntax**

```
void* FRPanelMgrCreatePanelEventHandler (
    FR\_PanelEventCallbacks callbacks
);
```

**Description**

Creates the pointer to the panel event handler.

**Parameter**


---

callbacks	[In] A callback set for the panel event handler.
-----------	--

---

**Return**

The pointer to the panel event handler.

**Head file reference**

fr\_appTempl.h: 3814

**Related method**

[FRPanelMgrCreate](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRPanelMgrDeletePanelMgr****Syntax**

```
void FRPanelMgrDeletePanelMgr (
    hParentWnd
);
```

**Description**

Deletes the panel manager.

**Parameter**

---

hParentWnd	[In] The input panel manager.
------------	-------------------------------

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 3803

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRPanelMgrDestroyPanelEventHandler****Syntax**

```
void FRPanelMgrDestroyPanelEventHandler (
    void* panelEventHandler
);
```

**Description**

Destroys the panel event handler.

**Parameter**

---

panelEventHandler	[In] The pointer to the panel event handler.
-------------------	--

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 3825

**Related method**[FRPanelMgrCreate](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRPanelMgrDockToFrameWindow****Syntax**

```
FS_BOOL FRPanelMgrDockToFrameWindow (
    FR\_PanelMgr panelMgr,
    FS\_DWORD dwAlignment
);
```

**Description**

Docks the panel to the frame window.

**Parameter**

panelMgr	[In] The input panel manager.
dwAlignment	[In] Specifies the alignment. References to MFC ControlBar styles.

**Return**

True for success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 3851

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRPanelMgrGetEnabledAlignment****Syntax**

```
FS_DWORD FRPanelMgrGetEnabledAlignment (
    FR\_PanelMgr panelMgr
);
```

**Description**

Gets the enabled alignment. References to MFC ControlBar styles.

**Parameter**

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

**Return**

The enabled alignment. References to MFC ControlBar styles.

**Head file reference**

fr\_appTempl.h: 3973

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRPanelMgrGetPanelCount****Syntax**

```
FS_INT32 FRPanelMgrGetPanelCount (
    FR\_PanelMgr panelMgr
);
```

**Description**

Gets the panel count.

**Parameter**


---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

**Return**

The panel count.

**Head file reference**

fr\_appTempl.h: 4008

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRPanelMgrGetPanelHwndByName****Syntax**

```
HWND FRPanelMgrGetPanelHwndByName (
    FR\_PanelMgr panelMgr,
    FS\_LPCSTR csName
);
```

**Description**

---

Gets the window handle to the panel by name.

#### Parameter

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

csName	[In] Specifies the name of the panel.
--------	---------------------------------------

---

#### Return

The window handle to the panel.

#### Head file reference

fr\_appTempl.h: 3916

#### Related method

##### Since

[SDK LATEEST VERSION > 7.1](#)

**FRPanelMgrGetPanelMgrFromChildFrm**

#### Syntax

```
FR_PanelMgr FRPanelMgrGetPanelMgrFromChildFrm (
    HWND hChildFrm
);
```

#### Description

Gets the specified panel manager.

#### Parameter

---

hChildFrm	[In] The specified child frame.
-----------	---------------------------------

---

#### Return

The specified panel manager.

#### Head file reference

fr\_appTempl.h: 3781

#### Related method

##### Since

[SDK LATEEST VERSION > 7.1](#)

**FRPanelMgrGetPanelNameByIndex**

**Syntax**

```
FS_LPCSTR FRPanelMgrGetPanelNameByIndex (
    FR\_PanelMgr panelMgr,
    FS\_INT32 nIndex
);
```

**Description**

Gets the specified panel name.

**Parameter**

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

nIndex	[In] Specifies the index of the panel.
--------	--

---

**Return**

The specified panel name.

**Head file reference**

fr\_appTempl.h: 3996

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRPanelMgrGetPanelTabNameByPt****Syntax**

```
FS_LPCSTR FRPanelMgrGetPanelTabNameByPt (
    FR\_PanelMgr panelMgr,
    FS\_INT32 x,
    FS\_INT32 y
);
```

**Description**

Gets the panel name by point.

**Parameter**

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

x	[In] The horizon position.
---	----------------------------

---

y	[In] The vertical position.
---	-----------------------------

---

**Return**

The panel name.

**Head file reference**

fr\_appTempl.h: 4033

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRPanelMgrGetPanelTabRectByName****Syntax**

```
void FRPanelMgrGetPanelTabRectByName (
    FR\_PanelMgr panelMgr,
    FS\_LPCSTR csName,
    FS\_Rect* outRtTab
);
```

**Description**

Gets the rectangle of the panel tab by name.

**Parameter**

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

csName	[In] The input panel name.
--------	----------------------------

---

outRtTab	[Out] It receives the rectangle of the panel tab.
----------	---

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 4046

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRPanelMgrGetPanelTabRectByPt****Syntax**

```
void FRPanelMgrGetPanelTabRectByPt (
    FR\_PanelMgr panelMgr,
```

---

```
FS_INT32 x,  
FS_INT32 y,  
FS_Rect* outRtTab  
);
```

**Description**

Gets the rectangle of the panel tab by point.

**Parameter**


---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

x	[In] The horizon position.
---	----------------------------

---

y	[In] The vertical position.
---	-----------------------------

---

outRtTab	[Out] It receives the rectangle of the panel tab.
----------	---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 4019

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRPanelMgrGetPanelViewByName****Syntax**

```
HWND FRPanelMgrGetPanelViewByName (  
    FR_PanelMgr panelMgr,  
    FS_LPCSTR csName  
)
```

**Description**

Gets the specified panel view handle.

**Parameter**


---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

csName	[In] Specifies the name of the panel.
--------	---------------------------------------

**Return**

The specified panel view handle.

**Head file reference**

fr\_appTempl.h: 3984

**Related method****Since**

[SDK LATEEST VERSION > 7.1](#)

**FRPanelMgrGetParentFrame****Syntax**

```
HWND FRPanelMgrGetParentFrame (
    FR\_PanelMgr panelMgr
);
```

**Description**

Gets the parent window handle.

**Parameter**

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

**Return**

The parent window handle.

**Head file reference**

fr\_appTempl.h: 3962

**Related method****Since**

[SDK LATEEST VERSION > 7.1](#)

**FRPanelMgrHasHistory****Syntax**

```
FS_BOOL FRPanelMgrHasHistory (
    FR\_PanelMgr panelMgr
);
```

**Description**

Checks whether the customer has configured the panel or not.

**Parameter**

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

**Return**

TRUE if the customer has configured the panel, otherwise not.

**Head file reference**

fr\_appTempl.h: 4071

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRPanelMgrHasPanelFloating****Syntax**

```
FS_BOOL FRPanelMgrHasPanelFloating (
    FR\_PanelMgr panelMgr
);
```

**Description**

Checks whether any panel is floating or not.

**Parameter**

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

**Return**

TRUE if any panel is floating, otherwise not.

**Head file reference**

fr\_appTempl.h: 3939

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FRPanelMgrHidePanelTabByName****Syntax**

```
void FRPanelMgrHidePanelTabByName (
    FR\_PanelMgr panelMgr,
    FS\_LPCSTR csName
);
```

**Description**

Hides the specified panel by name.

#### Parameter

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

csName	[In] The input panel name.
--------	----------------------------

---

#### Return

void.

#### Head file reference

fr\_appTempl.h: 4059

#### Related method

#### Since

[SDK LATEEST VERSION > 7.1](#)

## FRPanelMgrIsAllPanelHide

#### Syntax

```
FS_BOOL FRPanelMgrIsAllPanelHide (
    FR\_PanelMgr panelMgr
);
```

#### Description

Checks whether all the panels are hidden or not.

#### Parameter

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

#### Return

TRUE if all the panels are hidden, otherwise not.

#### Head file reference

fr\_appTempl.h: 3928

#### Related method

#### Since

[SDK LATEEST VERSION > 7.1](#)

## FRPanelMgrIsPanelFloating

**Syntax**

```
FS_BOOL FRPanelMgrIsPanelFloating (
    FR\_PanelMgr panelMgr,
    FS\_LPCSTR csName
);
```

**Description**

Checks whether the panel is floating or not.

**Parameter**

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

csName	[In] Specifies the name of the panel.
--------	---------------------------------------

---

**Return**

TRUE if the panel is floating.

**Head file reference**

fr\_appTempl.h: 4148

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 10.0](#)

**FRPanelMgrIsPanelHide****Syntax**

```
FS_BOOL FRPanelMgrIsPanelHide (
    FR\_PanelMgr panelMgr,
    FS\_LPCSTR csName
);
```

**Description**

Checks whether the specified panel is hidden or not.

**Parameter**

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

csName	[In] Specifies the name of the panel.
--------	---------------------------------------

---

**Return**

TRUE if the specified panel is hidden, otherwise not.

**Head file reference**

fr\_appTempl.h: 3950

**Related method****Since**[SDK LATEEST VERSION > 7.1](#)**FRPanelMgrIsPanelSpreadOut****Syntax**

```
FS_BOOL FRPanelMgrIsPanelSpreadOut (
    FR\_PanelMgr panelMgr,
    FS\_LPCSTR csName
);
```

**Description**

Checks whether the panel is spread out or not.

**Parameter**

panelMgr	[In] The input panel manager.
csName	[In] Specifies the name of the panel.

**Return**

TRUE if the panel is spread out.

**Head file reference**

fr\_appTempl.h: 4107

**Related method****Since**[SDK LATEEST VERSION > 8.2.1](#)**FRPanelMgrLockAllPanel****Syntax**

```
void FRPanelMgrLockAllPanel (
    FR\_PanelMgr panelMgr
);
```

**Description**

Locks all the panels.

**Parameter**

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 4082

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRPanelMgrNewPanelMgr****Syntax**

```
FR_PanelMgr FRPanelMgrNewPanelMgr (
    HWND hParentWnd
);
```

**Description**Creates an instance of the panel manager. You have to invoke [FRPanelMgrCreate](#) then.**Parameter**


---

hParentWnd	[In] The input parent window.
------------	-------------------------------

---

**Return**

The new panel manager.

**Head file reference**

fr\_appTempl.h: 3792

**Related method**[FRPanelMgrCreate](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRPanelMgrRedockAllFloatToInitial****Syntax**

```
void FRPanelMgrRedockAllFloatToInitial (
    FR_PanelMgr panelMgr
);
```

**Description**

All floating panel is restored to the left side.

#### Parameter

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

#### Return

void.

#### Head file reference

fr\_appTempl.h: 4137

#### Related method

##### Since

[SDK\\_LATEEST\\_VERSION > 10.0](#)

## FRPanelMgrResetAllPanels

#### Syntax

```
void FRPanelMgrResetAllPanels (
    FR\_PanelMgr panelMgr
);
```

#### Description

Resets all the panels.

#### Parameter

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

#### Return

void.

#### Head file reference

fr\_appTempl.h: 3905

#### Related method

##### Since

[SDK\\_LATEEST\\_VERSION > 7.1](#)

## FRPanelMgrShowAllPanel

#### Syntax

```
void FRPanelMgrShowAllPanel (
    FR\_PanelMgr panelMgr,
    FS\_BOOL bShow
```

);

**Description**

Shows all the panels or not.

**Parameter**

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

bShow	[In] Shows all the panels or not.
-------	-----------------------------------

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 3880

**Related method****Since**

[SDK LATEEST VERSION > 7.1](#)

**FRPanelMgrShowPanelByName****Syntax**

```
void FRPanelMgrShowPanelByName (
    FR\_PanelMgr panelMgr,
    FS\_LPCSTR csName,
    FS\_BOOL bShow
);
```

**Description**

Shows the specified panel or not.

**Parameter**

---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

csName	[In] Specifies the name of the panel.
--------	---------------------------------------

---

bShow	[In] Shows the panel or not.
-------	------------------------------

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 3892

**Related method****Since**[SDK LATEEST VERSION > 7.1](#)**FRPanelMgrShowPanelByName2****Syntax**

```
void FRPanelMgrShowPanelByName2 (
    FR\_PanelMgr panelMgr,
    FS\_LPCSTR csName,
    FS\_BOOL bShow,
    FS\_BOOL bActive
);
```

**Description**

Shows the specified panel or not.

**Parameter**

panelMgr	[In] The input panel manager.
----------	-------------------------------

csName	[In] Specifies the name of the panel.
--------	---------------------------------------

bShow	[In] Shows the panel or not.
-------	------------------------------

bActive	[In] Activates the panel or not.
---------	----------------------------------

**Return**

void.

**Head file reference**

fr\_appTempl.h: 4093

**Related method****Since**[SDK LATEEST VERSION > 8.2.1](#)**FRPanelMgrShrinkPanelByName****Syntax**

```
void FRPanelMgrShrinkPanelByName (
    FR\_PanelMgr panelMgr,
    FS\_LPCSTR csName
```

);

**Description**

Shrink the specified panel by name.

**Parameter**


---

panelMgr	[In] The input panel manager.
----------	-------------------------------

---

csName	[In] The input panel name.
--------	----------------------------

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 4160

**Related method****Since**

[SDK LATEEST VERSION > 10.0](#)

## FR\_ProfStore

**Description**

## FR\_PropertyTools

**[Return from Used by](#)****Description**

The property tools is used to set line color and opacity of the PDF object in classic mode.

**Returned from****[FRPropertyToolsGet](#)****Used by**

[FRPropertyToolsEnable](#)  
[FRPropertyToolsIsEnabled](#)  
[FRPropertyToolsIsVisible](#)  
[FRPropertyToolsSetColor](#)  
[FRPropertyToolsSetEvent](#)  
[FRPropertyToolsSetOpacity](#)  
[FRPropertyToolsShow](#)

## Callbacks

### Callbacks summary

#### [FRPropertyToolOnColorChanged](#)

It is called by Foxit Reader when the color setting is changed.

#### [FRPropertyToolOnOpacityChanged](#)

It is called by Foxit Reader when the opacity setting is changed.

#### [FRPropertyToolOnBeginScroll](#)

It is called by Foxit Reader when beginning scrolling.

#### [FRPropertyToolOnEndScroll](#)

It is called by Foxit Reader when ending scrolling.

#### [FRPropertyToolOnOpacityScroll](#)

It is called by Foxit Reader when the opacity setting is scrolling.

### Callbacks detail

#### FRPropertyToolOnColorChanged

##### Syntax

```
typedef void (*FRPropertyToolOnColorChanged)(  
    FS_LPVVOID clientData,  
    COLORREF color  
)
```

##### Description

It is called by Foxit Reader when the color setting is changed.

##### Parameter

clientData	[In] The user-supplied data.
------------	------------------------------

color	[In] The color value.
-------	-----------------------

##### Return

void

##### Head file reference

fr\_barExpT.h: 1278

##### Group

[FR\\_PropertyToolCallbacksRec](#)

##### Since

[SDK LATEEST VERSION > 2.0](#)

#### FRPropertyToolOnOpacityChanged

**Syntax**

```
typedef void (*FRPropertyToolOnOpacityChanged)(  
    FS\_LPVOID clientData,  
    FS\_INT32 nOpacity  
>);
```

**Description**

It is called by Foxit Reader when the opacity setting is changed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

nOpacity	[In] The opacity value.
----------	-------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1291

**Group**

[FR\\_PropertyToolCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRPropertyToolOnBeginScroll****Syntax**

```
typedef void (*FRPropertyToolOnBeginScroll)(  
    FS\_LPVOID clientData  
>);
```

**Description**

It is called by Foxit Reader when beginning scrolling.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1303

**Group**[FR\\_PropertyToolCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRPropertyToolOnEndScroll****Syntax**

```
typedef void (*FRPropertyToolOnEndScroll)(  
    FS\_LPVOID clientData  
)
```

**Description**

It is called by Foxit Reader when ending scrolling.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1315

**Group**[FR\\_PropertyToolCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRPropertyToolOnOpacityScroll****Syntax**

```
typedef void (*FRPropertyToolOnOpacityScroll)(  
    FS\_LPVOID clientData,  
    FS\_INT32 nOpacity  
)
```

**Description**

It is called by Foxit Reader when the opacity setting is scrolling.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

nOpacity	[In] The opacity value.
----------	-------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 1328

**Group**[FR\\_PropertyToolCallbacksRec](#)**Since**

SDK LATEEST VERSION &gt; 2.1.0.4

## Functions

### Functions summary

[\*\*FRPropertyToolsEnable\*\*](#)

Sets to enable the property tools or not.

[\*\*FRPropertyToolsGet\*\*](#)

Gets the property tools. The property tools is used to set line color and opacity of the PDF object in classic mode.

[\*\*FRPropertyToolsIsEnabled\*\*](#)

Checks whether the property tools are enabled or not.

[\*\*FRPropertyToolsIsVisible\*\*](#)

Checks whether the property tools are visible or not.

[\*\*FRPropertyToolsReleaseEvent\*\*](#)Releases the event handle returned by [FRPropertyToolsSetEvent](#) .[\*\*FRPropertyToolsSetColor\*\*](#)

Sets the color of the property tools.

[\*\*FRPropertyToolsSetEvent\*\*](#)

Sets the callback functions that are called by Foxit Reader when the events occur.

[\*\*FRPropertyToolsSetOpacity\*\*](#)

Sets the opacity of the property tools.

[\*\*FRPropertyToolsShow\*\*](#)

Sets to show the property tools or not.

### Functions detail

[\*\*FRPropertyToolsEnable\*\*](#)**Syntax**

```
void FRPropertyToolsEnable (
    FR\_PropertyTools propertyTools,
    FS\_BOOL bEnable
);
```

**Description**

---

Sets to enable the property tools or not.

#### Parameter

---

propertyTools	[In] The input property tools object.
bEnable	[In] Sets TRUE to enable the property tools, otherwise not.

---

#### Return

void

#### Head file reference

fr\_barTempl.h: 5817

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

## FRPropertyToolsGet

#### Syntax

```
FR_PropertyTools FRPropertyToolsGet (void );
```

#### Description

Gets the property tools. The property tools is used to set line color and opacity of the PDF object in classic mode.

#### Return

The property tools object.

#### Head file reference

fr\_barTempl.h: 5737

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

## FRPropertyToolsIsEnabled

#### Syntax

```
FS_BOOL FRPropertyToolsIsEnabled (
    FR\_PropertyTools propertyTools
);
```

#### Description

Checks whether the property tools are enabled or not.

#### Parameter

---

propertyTools	[In] The input property tools object.
---------------	---------------------------------------

---

#### Return

[TRUE](#) if the property tools are enabled.

#### Head file reference

fr\_barTempl.h: 5806

#### Related method

##### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRPropertyToolsIsVisible

#### Syntax

```
FS_BOOL FRPropertyToolsIsVisible (
    FR\_PropertyTools propertyTools
);
```

#### Description

Checks whether the property tools are visible or not.

#### Parameter

---

propertyTools	[In] The input property tools object.
---------------	---------------------------------------

---

#### Return

[TRUE](#) if the property tools are visible, otherwise not.

#### Head file reference

fr\_barTempl.h: 5783

#### Related method

##### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRPropertyToolsReleaseEvent

#### Syntax

```
void FRPropertyToolsReleaseEvent (
    void* eventHandle
);
```

**Description**

Releases the event handle returned by [FRPropertyToolsSetEvent](#) .

**Parameter**

---

eventHandle	[In] The input event handle returned by <a href="#">FRPropertyToolsSetEvent</a> .
-------------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 5829

**Related method**

[FRPropertyToolsSetEvent](#)

**Since**

[SDK LATEEST VERSION > 7.1](#)

**FRPropertyToolsSetColor****Syntax**

```
void FRPropertyToolsSetColor (
    FR\_PropertyTools propertyTools,
    COLORREF color
);
```

**Description**

Sets the color of the property tools.

**Parameter**

---

propertyTools	[In] The input property tools object.
---------------	---------------------------------------

---

---

color	[In] The input color of the property tools.
-------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5759

**Related method****Since**

## [SDK\\_LATEEST\\_VERSION > 2.0](#)

### FRPropertyToolsSetEvent

#### Syntax

```
void* FRPropertyToolsSetEvent (
    FR\_PropertyTools propertyTools,
    FR\_PropertyToolCallbacks callbacks
);
```

#### Description

Sets the callback functions that are called by Foxit Reader when the events occur.

#### Parameter

---

propertyTools	[In] The input property tools object.
callbacks	[In] The callback functions. They are called by Foxit Reader when the events occur.

---

#### Return

The returned value is reserved.

#### Head file reference

fr\_barTempl.h: 5747

#### Related method

#### Since

## [SDK\\_LATEEST\\_VERSION > 2.0](#)

### FRPropertyToolsSetOpacity

#### Syntax

```
void FRPropertyToolsSetOpacity (
    FR\_PropertyTools propertyTools,
    FS\_INT32 nOpacity
);
```

#### Description

Sets the opacity of the property tools.

#### Parameter

---

propertyTools	[In] The input property tools object.
---------------	---------------------------------------

---

---

nOpacity	[In] The input opacity of the property tools.
----------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5771

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRPropertyToolsShow****Syntax**

```
void FRPropertyToolsShow (
    FR\_PropertyTools propertyTools,
    FS\_BOOL bShow
);
```

**Description**

Sets to show the property tools or not.

**Parameter**


---

propertyTools	[In] The input property tools object.
---------------	---------------------------------------

---

bShow	[In] Indicates whether shows the property tools or not.
-------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5794

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FR\_ResourcePropertyBox****[Return from Used by](#)****Description**

The property box is used to edit the properties of objects, such as the annotations, the pages and so on. The Foxit Reader creates the property box so that many types of objects can reuse the same property box to edit the properties.

## Returned from

[FRResourcePropertyBoxGet](#)

## Used by

[FRResourcePropertyBoxClosePropertyBox](#)  
[FRResourcePropertyBoxGetPropertyPage](#)  
[FRResourcePropertyBoxGetSourceFunc](#)  
[FRResourcePropertyBoxGetSourceType](#)  
[FRResourcePropertyBoxGetWnd](#)  
[FRResourcePropertyBoxIsPropertyBoxVisible](#)  
[FRResourcePropertyBoxOpenPropertyBox](#)  
[FRResourcePropertyBoxRegisterPropertyPage](#)  
[FRResourcePropertyBoxRegisterPropertyPage2](#)  
[FRResourcePropertyBoxRegisterSourceType](#)  
[FRResourcePropertyBoxRegisterSourceType2](#)  
[FRResourcePropertyBoxSetCurrentPropertyPage](#)  
[FRResourcePropertyBoxUpdatePropertyBox](#)

## Definitions

### Definitions summary

#### [FR\\_SOURCE\\_TYPE\\_ANNOT](#)

The source type is annotation.

#### [FR\\_SOURCE\\_TYPE\\_BOOKMARK](#)

The source type is bookmark.

#### [FR\\_SOURCE\\_TYPE\\_PAGE](#)

The source type is page.

#### [FR\\_SOURCE\\_TYPE\\_PAGEOBJECTS](#)

The source type is page objects.

#### [FR\\_SOURCE\\_TYPE\\_TAG](#)

The source type is tag.

#### [FR\\_SOURCE\\_TYPE\\_UNKNOWN](#)

Unknown type.

#### [FR\\_SOURCE\\_TYPE\\_USER](#)

The source type is user.

### Definitions detail

#### [FR\\_SOURCE\\_TYPE\\_ANNOT](#)

##### **Syntax**

```
#define FR_SOURCE_TYPE_ANNOT 0
```

##### **Description**

The source type is annotation.

**Group**  
[FRSOURCETYPE](#)

**Head file reference**  
fr\_viewExpT.h: 340

## FR\_SOURCE\_TYPE\_BOOKMARK

**Syntax**  
`#define FR_SOURCE_TYPE_BOOKMARK 1`

**Description**  
The source type is bookmark.

**Group**  
[FRSOURCETYPE](#)

**Head file reference**  
fr\_viewExpT.h: 342

## FR\_SOURCE\_TYPE\_PAGE

**Syntax**  
`#define FR_SOURCE_TYPE_PAGE 2`

**Description**  
The source type is page.

**Group**  
[FRSOURCETYPE](#)

**Head file reference**  
fr\_viewExpT.h: 344

## FR\_SOURCE\_TYPE\_PAGEOBJECTS

**Syntax**  
`#define FR_SOURCE_TYPE_PAGEOBJECTS 3`

**Description**  
The source type is page objects.

**Group**  
[FRSOURCETYPE](#)

**Head file reference**

fr\_viewExpT.h: 346

## FR\_SOURCE\_TYPE\_TAG

### Syntax

#define FR\_SOURCE\_TYPE\_TAG 5

### Description

The source type is tag.

### Group

[FRSOURCETYPE](#)

### Head file reference

fr\_viewExpT.h: 350

## FR\_SOURCE\_TYPE\_UNKNOWN

### Syntax

#define FR\_SOURCE\_TYPE\_UNKNOWN -1

### Description

Unknown type.

### Group

[FRSOURCETYPE](#)

### Head file reference

fr\_viewExpT.h: 338

## FR\_SOURCE\_TYPE\_USER

### Syntax

#define FR\_SOURCE\_TYPE\_USER 4

### Description

The source type is user.

### Group

[FRSOURCETYPE](#)

### Head file reference

fr\_viewExpT.h: 348

## Callbacks

## Callbacks summary

### **FRResourcePropertyNotifyOnLockObjects**

It is called by foxit reader when the user clicks the check box to lock the source.

### **FRResourcePropertyPageSetPropertyBox**

It is called by foxit reader to pass the property box to the plug-in.

### **FRResourcePropertyPageGetPageTitle**

It is called by foxit reader to get the title of the property page.

### **FRResourcePropertyPageGetPageName**

It is called by foxit reader to get the name of the property page.

### **FRResourcePropertyPageGetSequenceNumber**

It is called by foxit reader to get the sequence number of the property page.

### **FRResourcePropertyPageIsSourceSupported**

It is called by foxit reader to check whether the source is supported.

### **FRResourcePropertyPageUpdatePage**

It is called by foxit reader to notify the plug-in to update the property page.

### **FRResourcePropertyPageGetPageHWnd**

It is called by foxit reader to get the window handle of the property page.

### **FRResourcePropertyPageGetPageSize**

It is called by foxit reader to get the size of the property page.

### **FRResourcePropertyPageCreatePage**

It is called by foxit reader to notify the plug-in to create the property page.

### **FRResourcePropertyPageOnDeactive**

It is called by foxit reader to notify the plug-in to deactivate the property page.

### **FRResourcePropertyPageOnLangUIChange**

It is called by foxit reader to notify the plug-in to change the language of the property page.

### **FRResourcePropertySourceGetFRDDocument**

It is called by foxit reader to get the [FR\\_Document](#) object.

### **FRResourcePropertySourceCountSelectObjects**

It is called by foxit reader to get the count of the selected objects.

### **FRResourcePropertySourceGetSelectObject**

It is called by foxit reader to get the selected object by index.

### **FRResourcePropertySourceGetSelectObjectType**

It is called by foxit reader to get the types of the selected objects.

### **FRResourcePropertySourceIsLockedObjectExisted**

It is called by foxit reader to check whether the locked object exists.

### **FRResourcePropertySourceIsDisabledObjectExisted**

It is called by foxit reader to check whether the disabled object exists.

### **FRResourcePropertySourceIsPropertyBoxEnabled**

It is called by foxit reader to check whether the property box is enabled.

### **FRResourcePropertySourceGetPropertyBoxTitle**

It is called by foxit reader to get the title of the property box.

### **FRResourcePropertySourceCanExistedSetDefaultButton**

It is called by foxit reader to determine whether to show the check box to set current properties as default.

### **FRResourcePropertySourceOnBoxClose**

It is called by foxit reader when the property box is closed.

## Callbacks detail

### **FRResourcePropertyNotifyOnLockObjects**

**Syntax**

```
typedef void (*FRResourcePropertyNotifyOnLockObjects)(  
    FS_LPVVOID clientData,  
    FS_BOOL bLocked  
) ;
```

**Description**

It is called by foxit reader when the user clicks the check box to lock the source.

**Parameter**

clientData	[In] The user-supplied data.
bLocked	[In] Indicates whether to lock the source or not.

**Return**

void

**Head file reference**

fr\_viewExpT.h: 381

**Group**

[FR\\_ResourcePropertyNotifyCallbacksRec](#)

**Related method**

[FRResourcePropertyBoxRegisterSourceType](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRResourcePropertyPageSetPropertyBox****Syntax**

```
typedef void (*FRResourcePropertyPageSetPropertyBox)(  
    FS_LPVVOID clientData,  
    FR_ResourcePropertyBox pPropertyBox  
) ;
```

**Description**

It is called by foxit reader to pass the property box to the plug-in.

**Parameter**

clientData	[In] The user-supplied data.
pPropertyBox	[In] The property box.

**Return**

void

**Head file reference**

fr\_viewExpT.h: 542

**Group**

[FR\\_ResourcePropertyPageCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRResourcePropertyPageGetPageTitle

**Syntax**

```
typedef void (*FRResourcePropertyPageGetPageTitle)(  
    FS\_LPVOID clientData,  
    FS\_WideString outTitle  
)
```

**Description**

It is called by foxit reader to get the title of the property page.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

outTitle	[Out] It receives the title of the property page.
----------	---

---

**Return**

void

**Head file reference**

fr\_viewExpT.h: 554

**Group**

[FR\\_ResourcePropertyPageCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRResourcePropertyPageGetPageName

**Syntax**

```
typedef void (*FRResourcePropertyPageGetPageName)(  
    FS\_LPVOID clientData,  
    FS\_WideString outName
```

);

**Description**

It is called by foxit reader to get the name of the property page.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

outName	[Out] It receives the name of the property page.
---------	--

---

**Return**

void

**Head file reference**

fr\_viewExpT.h: 566

**Group**

[FR\\_ResourcePropertyPageCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRResourcePropertyPageGetSequenceNumber

**Syntax**

```
typedef FS_INT32 (*FRResourcePropertyPageGetSequenceNumber)(  
    FS\_LPVOID clientData  
)
```

**Description**

It is called by foxit reader to get the sequence number of the property page.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

The sequence number of the property page.

**Head file reference**

fr\_viewExpT.h: 577

**Group**

[FR\\_ResourcePropertyPageCallbacksRec](#)

**Since**  
[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRResourcePropertyPageIsSourceSupported

### Syntax

```
typedef FS_BOOL (*FRResourcePropertyPageIsSourceSupported)(  
    FS_LPVVOID clientData,  
    FS_INT32 nSource,  
    FR_ResourcePropertySource pSourceFunc  
);
```

### Description

It is called by foxit reader to check whether the source is supported.

### Parameter

clientData	[In] The user-supplied data.
nSource	[In] The source type.
pSourceFunc	[In] The property source.

### Return

[TRUE](#) if the souce is supported, otherwise not.

### Head file reference

fr\_viewExpT.h: 590

### Group

[FR\\_ResourcePropertyPageCallbacksRec](#)

### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRResourcePropertyPageUpdatePage

### Syntax

```
typedef void (*FRResourcePropertyPageUpdatePage)(  
    FS_LPVVOID clientData  
);
```

### Description

It is called by foxit reader to notify the plug-in to update the property page.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void

**Head file reference**

fr\_viewExpT.h: 601

**Group**[FR\\_ResourcePropertyPageCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRResourcePropertyPageGetPageHWnd****Syntax**

```
typedef HWND (*FRResourcePropertyPageGetPageHWnd)(  
    FS_LPVVOID clientData  
)
```

**Description**

It is called by foxit reader to get the window handle of the property page.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

The window handle of the property page.

**Head file reference**

fr\_viewExpT.h: 612

**Group**[FR\\_ResourcePropertyPageCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRResourcePropertyPageGetPageSize****Syntax**

```
typedef FS_BOOL (*FRResourcePropertyPageGetPageSize)(  
    FS_LPVVOID clientData,  
    FS_INT32* outCX,  
    FS_INT32* outCY  
)
```

**Description**

It is called by foxit reader to get the size of the property page.

**Parameter**

clientData	[In] The user-supplied data.
outCX	[Out] It receives the width of the property page.
outCY	[Out] It receives the height of the property page.

**Return**

[TRUE](#) for success, otherwise the property page will use the default size.

**Head file reference**

fr\_viewExpT.h: 625

**Group**

[FR\\_ResourcePropertyPageCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRResourcePropertyPageCreatePage****Syntax**

```
typedef FS_BOOL (*FRResourcePropertyPageCreatePage)(  
    FS\_LPVVOID clientData,  
    HWND hParent  
)
```

**Description**

It is called by foxit reader to notify the plug-in to create the property page.

**Parameter**

clientData	[In] The user-supplied data.
hParent	[In] The parent window handle.

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

fr\_viewExpT.h: 637

**Group**

[FR\\_ResourcePropertyPageCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRResourcePropertyPageOnDeactive

**Syntax**

```
typedef FS_BOOL (*FRResourcePropertyPageOnDeactive)(  
    FS\_LPVVOID clientData  
);
```

**Description**

It is called by foxit reader to notify the plug-in to deactivate the property page.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

fr\_viewExpT.h: 648

**Group**

[FR\\_ResourcePropertyPageCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRResourcePropertyPageOnLangUIChange

**Syntax**

```
typedef void (*FRResourcePropertyPageOnLangUIChange)(  
    FS\_LPVVOID clientData  
);
```

**Description**

It is called by foxit reader to notify the plug-in to change the language of the property page.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void

**Head file reference**

fr\_viewExpT.h: 659

**Group**[FR\\_ResourcePropertyPageCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRResourcePropertySourceGetFRDocument****Syntax**

```
typedef FR_Document (*FRResourcePropertySourceGetFRDocument)(
    FS_LPVVOID clientData
);
```

**Description**It is called by foxit reader to get the [FR\\_Document](#) object.**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**The [FR\\_Document](#) object.**Head file reference**

fr\_viewExpT.h: 408

**Group**[FR\\_ResourcePropertySourceCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRResourcePropertySourceCountSelectObjects****Syntax**

```
typedef FS_INT32 (*FRResourcePropertySourceCountSelectObjects)(
    FS_LPVVOID clientData
);
```

**Description**

It is called by foxit reader to get the cout of the selected objects.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

The cout of the selected objects.

**Head file reference**

fr\_viewExpT.h: 419

**Group**

[FR\\_ResourcePropertySourceCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

## FRResourcePropertySourceGetSelectObject

**Syntax**

```
typedef void* (*FRResourcePropertySourceGetSelectObject)(  
    FS_LPVVOID clientData,  
    FS_INT32 index  
)
```

**Description**

It is called by foxit reader to get the selected object by index.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

index	[In] The specified index.
-------	---------------------------

---

**Return**

The selected object.

**Head file reference**

fr\_viewExpT.h: 431

**Group**

[FR\\_ResourcePropertySourceCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

## FRResourcePropertySourceGetSelectObjectType

### Syntax

```
typedef FS_INT32 (*FRResourcePropertySourceGetSelectObjectType)(  
    FS_LPVVOID clientData,  
    FS_ByteStringArray outArray  
);
```

### Description

It is called by foxit reader to get the types of the selected objects.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

outArray	[Out] It receives the types.
----------	------------------------------

---

### Return

The returned value is reserved.

### Head file reference

fr\_viewExpT.h: 443

### Group

[FR\\_ResourcePropertySourceCallbacksRec](#)

### Since

[SDK LATEEST VERSION > 2.0](#)

## FRResourcePropertySourceIsLockedObjectExisted

### Syntax

```
typedef FS_BOOL (*FRResourcePropertySourceIsLockedObjectExisted)(  
    FS_LPVVOID clientData  
);
```

### Description

It is called by foxit reader to check whether the locked object exists.

### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

### Return

[TRUE](#) if the locked object exists, otherwise not.

**Head file reference**

fr\_viewExpT.h: 454

**Group**[FR\\_ResourcePropertySourceCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRResourcePropertySourceIsDisabledObjectExisted****Syntax**

```
typedef FS_BOOL (*FRResourcePropertySourceIsDisabledObjectExisted)(  
    FS\_LPVOID clientData  
);
```

**Description**

It is called by foxit reader to check whether the disabled object exists.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

**Return**

[TRUE](#) if the disabled object exists, otherwise not.

**Head file reference**

fr\_viewExpT.h: 465

**Group**[FR\\_ResourcePropertySourceCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRResourcePropertySourceIsPropertyBoxEnabled****Syntax**

```
typedef FS_BOOL (*FRResourcePropertySourceIsPropertyBoxEnabled)(  
    FS\_LPVOID clientData,  
    FS\_LPCSTR lpsObjectType  
);
```

**Description**

It is called by foxit reader to check whether the property box is enabled.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

lpsObjectType	[In] The specified object type.
---------------	---------------------------------

---

**Return**

[TRUE](#) if the property box is enabled, otherwise not.

**Head file reference**

fr\_viewExpT.h: 477

**Group**

[FR\\_ResourcePropertySourceCallbacksRec](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRResourcePropertySourceGetPropertyBoxTitle****Syntax**

```
typedef void (*FRResourcePropertySourceGetPropertyBoxTitle)(  
    FS\_LPVOID clientData,  
    FS\_LPCSTR lpsObjectType,  
    FS\_WideString outTitle  
);
```

**Description**

It is called by foxit reader to get the title of the property box.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

lpsObjectType	[In] The specified object type.
---------------	---------------------------------

---

outTitle	[Out] It receives the title of the property box.
----------	--

---

**Return**

void

**Head file reference**

fr\_viewExpT.h: 490

**Group**

[FR\\_ResourcePropertySourceCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRResourcePropertySourceCanExistedSetDefaultButton

**Syntax**

```
typedef FS_BOOL (*FRResourcePropertySourceCanExistedSetDefaultButton)(  
    FS_LPVOID clientData  
);
```

**Description**

It is called by foxit reader to determine whether to show the check box to set current properties as default.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

[TRUE](#) if the check box is to be shown, otherwise not.

**Head file reference**

fr\_viewExpT.h: 501

**Group**

[FR\\_ResourcePropertySourceCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRResourcePropertySourceOnBoxClose

**Syntax**

```
typedef void (*FRResourcePropertySourceOnBoxClose)(  
    FS_LPVOID clientData,  
    FS_INT32 nType,  
    FS_INT32 bSetDefaultButton  
);
```

**Description**

It is called by foxit reader when the property box is closed.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

nType	[In] The source type.
bSetDefaultButton	[In] Whether the check box is checked or not.

---

**Return**

void

**Head file reference**

fr\_viewExpT.h: 514

**Group**[FR\\_ResourcePropertySourceCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)

## Functions

### Functions summary

[\*\*FRResourcePropertyBoxClosePropertyBox\*\*](#)

Closes the property box.

[\*\*FRResourcePropertyBoxDestroyPage\*\*](#)Destroys the page object returned by [FRResourcePropertyBoxRegisterPropertyPage2](#) .[\*\*FRResourcePropertyBoxDestroySource\*\*](#)Destroys the source object returned by [FRResourcePropertyBoxRegisterSourceType2](#) .[\*\*FRResourcePropertyBoxGet\*\*](#)

Gets the property box. The Foxit Reader creates the property box so that many types of objects can reuse the same property box to edit the properties. For example, the objects can be annotations, pages and so on.

[\*\*FRResourcePropertyBoxGetPropertyPage\*\*](#)

Gets the property page by name.

[\*\*FRResourcePropertyBoxGetSourceFunc\*\*](#)

Gets the specified property source.

[\*\*FRResourcePropertyBoxGetSourceType\*\*](#)

Gets the source type of the property box.

[\*\*FRResourcePropertyBoxGetWnd\*\*](#)

Gets the window handle of the property box.

[\*\*FRResourcePropertyBoxIsPropertyBoxVisible\*\*](#)

Checks whether the property box is visible or not.

[\*\*FRResourcePropertyBoxOpenPropertyBox\*\*](#)

Opens the property box.

[\*\*FRResourcePropertyBoxRegisterPropertyPage\*\*](#)

Registers the callbacks used to add the new property page to the property box.

[\*\*FRResourcePropertyBoxRegisterPropertyPage2\*\*](#)

Registers the callbacks used to add the new property page to the property box.

[\*\*FRResourcePropertyBoxRegisterSourceType\*\*](#)

Registers the source type.

**[FRResourcePropertyBoxRegisterSourceType2](#)**

Registers the source type.

**[FRResourcePropertyBoxSetCurrentPropertyPage](#)**

Sets the current property page.

**[FRResourcePropertyBoxUpdatePropertyBox](#)**

Updates the property box.

## Functions detail

**[FRResourcePropertyBoxClosePropertyBox](#)****Syntax**

```
void FRResourcePropertyBoxClosePropertyBox (
    FR\_ResourcePropertyBox rpBox
);
```

**Description**

Closes the property box.

**Parameter**

---

rpBox	[In] The input property box object.
-------	-------------------------------------

---

**Return**

void

**Head file reference**

fr\_viewTempl.h: 1166

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**[FRResourcePropertyBoxDestroyPage](#)****Syntax**

```
void FRResourcePropertyBoxDestroyPage (
    void* pPage
);
```

**Description**

Destroys the page object returned by [FRResourcePropertyBoxRegisterPropertyPage2](#).

**Parameter**

---

pPage	[In] The page object returned by <a href="#">FRResourcePropertyBoxRegisterPropertyPage2</a> .
-------	--

---

**Return**

void.

**Head file reference**

fr\_viewTempl.h: 1227

**Related method****Since**

[SDK LATEEST VERSION > 3.0](#)

**FRResourcePropertyBoxDestroySource****Syntax**

```
void FRResourcePropertyBoxDestroySource (  
    pPage  
) ;
```

**Description**

Destroys the source object returned by [FRResourcePropertyBoxRegisterSourceType2](#) .

**Parameter**

---

pPage	[In] The source object returned by <a href="#">FRResourcePropertyBoxRegisterSourceType2</a> .
-------	--

---

**Return**

void.

**Head file reference**

fr\_viewTempl.h: 1253

**Related method****Since**

[SDK LATEEST VERSION > 3.0](#)

**FRResourcePropertyBoxGet****Syntax**

```
FR_ResourcePropertyBox FRResourcePropertyBoxGet (void );
```

**Description**

Gets the property box. The Foxit Reader creates the property box so that many types of objects can reuse the same property box to edit the properties. For example, the objects can be annotations, pages and so on.

**Return**

The property box. It is used to edit the properties of objects, such as the annotations, the pages and so on.

**Head file reference**

fr\_viewTempl.h: 1081

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRResourcePropertyBoxGetPropertyPage****Syntax**

```
FR_ResourcePropertyPage FRResourcePropertyBoxGetPropertyPage (
    FR_ResourcePropertyBox rpBox,
    FS_LPCWSTR lpwsName
);
```

**Description**

Gets the property page by name.

**Parameter**

rpBox	[In] The input property box object.
-------	-------------------------------------

lpwsName	[In] The specified name of the property page.
----------	---

**Return**

The property page.

**Head file reference**

fr\_viewTempl.h: 1199

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRResourcePropertyBoxGetSourceFunc****Syntax**

```
FR_ResourcePropertySource FRResourcePropertyBoxGetSourceFunc (
    FR_ResourcePropertyBox rpBox,
    FS_INT32 nSource
);
```



**Description**

Gets the specified property source.

**Parameter**

---

rpBox	[In] The input property box object.
nSource	[In] The specified source type. See the definitions of <i>FRSOURCETYPE</i> .

---

**Return**

The property source.

**Head file reference**

fr\_viewTempl.h: 1130

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRResourcePropertyBoxGetSourceType

**Syntax**

```
FS_INT32 FRResourcePropertyBoxGetSourceType (
    FR\_ResourcePropertyBox rpBox
);
```

**Description**

Gets the source type of the property box.

**Parameter**

---

rpBox	[In] The input property box object.
-------	-------------------------------------

---

**Return**

The source type. See the definitions of *FRSOURCETYPE* .

**Head file reference**

fr\_viewTempl.h: 1119

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRResourcePropertyBoxGetWnd

### Syntax

```
HWND FRResourcePropertyBoxGetWnd (
    FR\_ResourcePropertyBox rpBox
);
```

### Description

Gets the window handle of the property box.

### Parameter

---

rpBox	[In] The input property box object.
-------	-------------------------------------

---

### Return

The window handle of the property box.

### Head file reference

fr\_viewTempl.h: 1188

### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRResourcePropertyBoxIsPropertyBoxVisible

### Syntax

```
FS_BOOL FRResourcePropertyBoxIsPropertyBoxVisible (
    FR\_ResourcePropertyBox rpBox
);
```

### Description

Checks whether the property box is visible or not.

### Parameter

---

rpBox	[In] The input property box object.
-------	-------------------------------------

---

### Return

[TRUE](#) if the property box is visible, otherwise not.

### Head file reference

fr\_viewTempl.h: 1177

### Related method

#### Since



[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRResourcePropertyBoxOpenPropertyBox****Syntax**

```
void FRResourcePropertyBoxOpenPropertyBox (
    FR\_ResourcePropertyBox rpBox,
    FS\_INT32 nSource
);
```

**Description**

Opens the property box.

**Parameter**

rpBox	[In] The input property box object.
nSource	[In] The specified source type. See the definitions of <i>FRSOURCETYPE</i> .

**Return**

void

**Head file reference**

fr\_viewTempl.h: 1154

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRResourcePropertyBoxRegisterPropertyPage****Syntax**

```
void FRResourcePropertyBoxRegisterPropertyPage (
    FR\_ResourcePropertyBox rpBox,
    FR\_ResourcePropertyPageCallbacks pPage
);
```

**Description**

Registers the callbacks used to add the new property page to the property box.

**Parameter**

rpBox	[In] The input property box object.
-------	-------------------------------------

pPage	[In] The input callbacks used to add the new property page to the property box.
-------	---

---

**Return**  
void

**Head file reference**  
fr\_viewTempl.h: 1092

**Related method**

**Since**  
[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRResourcePropertyBoxRegisterPropertyPage2

**Syntax**

```
void* FRResourcePropertyBoxRegisterPropertyPage2 (
    FR\_ResourcePropertyBox rpBox,
    FR\_ResourcePropertyPageCallbacks pPage
);
```

**Description**

Registers the callbacks used to add the new property page to the property box.

**Parameter**

---

rpBox	[In] The input property box object.
-------	-------------------------------------

pPage	[In] The input callbacks used to add the new property page to the property box.
-------	---

---

**Return**

The page object that needs to be destroyed by [FRResourcePropertyBoxDestroyPage](#).

**Head file reference**

fr\_viewTempl.h: 1223

**Related method**

**Since**  
[SDK\\_LATEEST\\_VERSION > 3.0](#)

## FRResourcePropertyBoxRegisterSourceType

**Syntax**

```
void FRResourcePropertyBoxRegisterSourceType (
    FR\_ResourcePropertyBox rpBox,
```



---

```
FS_INT32 nSource,
FS_BOOL bLockButton,
FR_ResourcePropertySourceCallbacks pSourceFunc,
FR_ResourcePropertyNotifyCallbacks pNotifyFunc
);
```

**Description**

Registers the source type.

**Parameter**

rpBox	[In] The input property box object.
nSource	[In] The input source type. See the definitions of <i>FRSOURCETYPE</i> .
bLockButton	[In] Indicates whether to show the check box used to lock the source.
pSourceFunc	[In] The input callbacks used to deal with the source of the property box.
pNotifyFunc	[In] The input callbacks that will be called when events occur.

**Return**

void

**Head file reference**

fr\_viewTempl.h: 1104

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRResourcePropertyBoxRegisterSourceType2****Syntax**

```
void* FRResourcePropertyBoxRegisterSourceType2 (
    FR_ResourcePropertyBox rpBox,
    FS_INT32 nSource,
    FS_BOOL bLockButton,
    FR_ResourcePropertySourceCallbacks pSourceFunc,
    FR_ResourcePropertyNotifyCallbacks pNotifyFunc
);
```

**Description**

Registers the source type.

#### Parameter

rpBox	[In] The input property box object.
nSource	[In] The input source type. See the definitions of <i>FRSOURCETYPE</i> .
bLockButton	[In] Indicates whether to show the check box used to lock the source.
pSourceFunc	[In] The input callbacks used to deal with the source of the property box.
pNotifyFunc	[In] The input callbacks that will be called when events occur.

#### Return

The source object that needs to be destroyed by [FRResourcePropertyBoxDestroySource](#) .

#### Head file reference

fr\_viewTempl.h: 1246

#### Related method

#### Since

[SDK LATEEST VERSION > 2.0](#)

## FRResourcePropertyBoxSetCurrentPropertyPage

#### Syntax

```
void FRResourcePropertyBoxSetCurrentPropertyPage (
    FR\_ResourcePropertyBox rpBox,
    FR\_ResourcePropertyPage pPage
);
```

#### Description

Sets the current property page.

#### Parameter

rpBox	[In] The input property box object.
pPage	[In] The input property page.

#### Return

void

**Head file reference**

fr\_viewTempl.h: 1211

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRResourcePropertyBoxUpdatePropertyBox

**Syntax**

```
void FRResourcePropertyBoxUpdatePropertyBox (
    FR\_ResourcePropertyBox rpBox,
    FS\_INT32 nSource
);
```

**Description**

Updates the property box.

**Parameter**

rpBox	[In] The input property box object.
nSource	[In] The specified source type. See the definitions of <i>FRSOURCETYPE</i> .

**Return**

void

**Head file reference**

fr\_viewTempl.h: 1142

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FR\_ResourcePropertyPage

**[Return from Used by](#)****Description**

The property page of the property box.

Returned from

[\*\*FRResourcePropertyBoxGetPropertyPage\*\*](#)

Used by

[\*\*FRResourcePropertyBoxRegisterPropertyPage\*\*](#)

[\*\*FRResourcePropertyBoxRegisterPropertyPage2\*\*](#)

[\*\*FRResourcePropertyBoxSetCurrentPropertyPage\*\*](#)

## FR\_ResourcePropertySource

[\*\*Return from\*\*](#) [\*\*Used by\*\*](#)

Description

The property source of the property box.

Returned from

[\*\*FRResourcePropertyBoxGetSourceFunc\*\*](#)

Used by

[\*\*FRResourcePropertyBoxRegisterSourceType\*\*](#)

[\*\*FRResourcePropertyBoxRegisterSourceType2\*\*](#)

## FR\_RibbonBackStageViewItem

[\*\*Return from\*\*](#) [\*\*Used by\*\*](#)

Description

The [\*\*FR\\_RibbonBackStageViewItem\*\*](#) object can be added under FILE category. The back stage view item is associated with a view. You can add your own dialog or property sheet page on the view. See [\*\*FRRibbonBackStageViewItemAddDialog\*\*](#) d [\*\*FRRibbonBackStageViewItemAddPropertySheetPage\*\*](#) .

Returned from

[\*\*FRRibbonBarAddBackStageViewItem\*\*](#)

[\*\*FRRibbonBarGetBackStageViewItem\*\*](#)

Used by

[\*\*FRRibbonBackStageViewItemAddDialog\*\*](#)

[\*\*FRRibbonBackStageViewItemAddPropertySheetGroup\*\*](#)

[\*\*FRRibbonBackStageViewItemAddPropertySheetPage\*\*](#)

[\*\*FRRibbonBackStageViewItemBCGPReTranslateMessage\*\*](#)

[\*\*FRRibbonBackStageViewItemEnablePropertyPageRemove\*\*](#)

[\*\*FRRibbonBackStageViewItemGetElement\*\*](#)

[\*\*FRRibbonBackStageViewItemIsDialogExist\*\*](#)

---

[\*\*FRRibbonBackStageViewItemIsPropertySheetPageExist\*\*](#)  
[\*\*FRRibbonBackStageViewItemRedrawPropertySheet\*\*](#)  
[\*\*FRRibbonBackStageViewItemRemovePropertySheetPage\*\*](#)  
[\*\*FRRibbonBackStageViewItemSetDisable\*\*](#)  
[\*\*FRRibbonBackStageViewItemSetGroupTitle\*\*](#)  
[\*\*FRRibbonBackStageViewItemSetPageTitle\*\*](#)  
[\*\*FRRibbonBackStageViewItemSetPropertyActivePage\*\*](#)  
[\*\*FRRibbonBackStageViewItemSetTitle\*\*](#)  
[\*\*FRRibbonBackStageViewItemSetVisible\*\*](#)

## Enumerations

### Enumerations summary

#### [\*\*FRRibbonBackStageViewItemXMoveType\*\*](#)

The move type of the back stage view item.

#### [\*\*FRRibbonBackStageViewItemXSizeType\*\*](#)

The size type of the back stage view item.

### Enumerations detail

#### FRRibbonBackStageViewItemXMoveType

##### Syntax

```
enum FRRibbonBackStageViewItemXMoveType{
    FRRIBBONBACKSTAGEVIEWITEM\_MOVETYPE\_NONE,
    FRRIBBONBACKSTAGEVIEWITEM\_MOVETYPE\_HORZ,
    FRRIBBONBACKSTAGEVIEWITEM\_MOVETYPE\_VERT,
    FRRIBBONBACKSTAGEVIEWITEM\_MOVETYPE\_BOTH /\* Moves in
both horizon and vertical direction. \*/
};
```

##### Description

The move type of the back stage view item.

##### Head file reference

fr\_barExpT.h: 663

#### [\*\*FRRIBBONBACKSTAGEVIEWITEM\\_MOVETYPE\\_NONE\*\*](#)

Unknown type.

#### [\*\*FRRIBBONBACKSTAGEVIEWITEM\\_MOVETYPE\\_HORZ\*\*](#)

Moves in horizon direction.

#### [\*\*FRRIBBONBACKSTAGEVIEWITEM\\_MOVETYPE\\_VERT\*\*](#)

Moves in vertical direction.

#### [\*\*FRRIBBONBACKSTAGEVIEWITEM\\_MOVETYPE\\_BOTH\*\* /\\* Moves in both horizon and vertical direction. \\*/](#)

#### FRRibbonBackStageViewItemXSizeType

**Syntax**

```
enum FRRibbonBackStageViewItemXSizeType{
    FRRIBBONBACKSTAGEVIEWITEM_SIZETYPE_NONE,
    FRRIBBONBACKSTAGEVIEWITEM_SIZETYPE_HORZ,
    FRRIBBONBACKSTAGEVIEWITEM_SIZETYPE_VERT,
    FRRIBBONBACKSTAGEVIEWITEM_SIZETYPE_BOTH /* The size in
both horizon and vertical direction. */
};
```

**Description**

The size type of the back stage view item.

**Head file reference**

fr\_barExpT.h: 674

**FRRIBBONBACKSTAGEVIEWITEM\_SIZETYPE\_NONE**

Unknown type.

**FRRIBBONBACKSTAGEVIEWITEM\_SIZETYPE\_HORZ**

The size in horizon direction.

**FRRIBBONBACKSTAGEVIEWITEM\_SIZETYPE\_VERT**

The size in vertical direction.

**FRRIBBONBACKSTAGEVIEWITEM\_SIZETYPE\_BOTH** /\* The size in both horizon and vertical direction. \*/**Callbacks****Callbacks summary****FRRibbonBackstageCreateProc**

The callback function is called to notify the plug-in to create the dialog when the back stage view is shown.

**FRRibbonBackstageDestoryProc**

The callback function is called to notify the plug-in to destroy the dialog when the back stage view is closed.

**FRRibbonBackstageSelectPageProc**

The callback function is called to notify the plug-in that the specified property page is selected.

**Callbacks detail****FRRibbonBackstageCreateProc****Syntax**

```
typedef HWND (*FRRibbonBackstageCreateProc)(
    HWND hParentWnd,
    void* clientData
);
```

**Description**

The callback function is called to notify the plug-in to create the dialog when the back stage view is shown.

**Parameter**


---

hParentWnd	[In] The parent window handle.
------------	--------------------------------

---

clientData	[In] The client data.
------------	-----------------------

---

**Return**

The dialog handle created.

**Head file reference**

fr\_barExpT.h: 631

**Related method**

[FRRibbonBackStageViewItemAddDialog](#)

[FRRibbonBackStageViewItemAddPropertySheetPage](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonBackstageDestoryProc****Syntax**

```
typedef void (*FRRibbonBackstageDestoryProc)(  
    void* pDialog,  
    void* clientData  
)
```

**Description**

The callback function is called to notify the plug-in to destroy the dialog when the back stage view is closed.

**Parameter**


---

pDialog	[In] The dialog handle to be destroyed.
---------	---

---

clientData	[In] The client data.
------------	-----------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 645

**Related method**[FRRibbonBackStageViewItemAddDialog](#)[FRRibbonBackStageViewItemAddPropertySheetPage](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonBackstageSelectPageProc****Syntax**

```
typedef void (*FRRibbonBackstageSelectPageProc)(  
    FS LPCSTR lpsPageName,  
    HWND hWnd  
);
```

**Description**

The callback function is called to notify the plug-in that the specified property page is selected.

**Parameter**

<code>lpsPageName</code>	[In] The specified property page of the back stage view item.
--------------------------	---

<code>hWnd</code>	[In] The window handle of the specified property page of the back stage view item.
-------------------	--

**Return**

void

**Head file reference**

fr\_barExpT.h: 658

**Related method**[FRRibbonBackStageViewItemSetPageSelectProc](#)**Since**[SDK LATEEST VERSION > 7.3.1](#)

## Functions

### Functions summary

**[FRRibbonBackStageViewItemAddAnchor](#)**

This interface is used to adjust the control position of the dialog added on the back stage view.

**[FRRibbonBackStageViewItemAddDialog](#)**

You can create your own dialog on the back stage view when it is shown. You can either add a dialog or a property sheet on the back stage view.

**FRRibbonBackStageViewItemAddPropertySheetGroup**

Adds a new property sheet group.

**FRRibbonBackStageViewItemAddPropertySheetPage**

You can create your own property sheet page on the back stage view when it is shown. You can either add a dialog or a property sheet on the back stage view.

**FRRibbonBackStageViewItemBCGPreTranslateMessage**

Sets the callback function called to notify the plug-in that the specified property page is selected.

**FRRibbonBackStageViewItemEnablePropertyPageRemove**

Sets whether the property page can be removed or not.

**FRRibbonBackStageViewItemGetElement**

Gets the ribbon element associated with the back stage view item. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**FRRibbonBackStageViewItemIsDialogExist**

Checks whether the dialog exists or not.

**FRRibbonBackStageViewItemIsPropertySheetPageExist**

Checks whether the property sheet page exists or not.

**FRRibbonBackStageViewItemModifiedToRibbonStyleButton**

Modifies the MFC controls to ribbon style button.

**FRRibbonBackStageViewItemRedrawPropertySheet**

Redraws the property sheet.

**FRRibbonBackStageViewItemRemovePropertySheetPage**

Removes the specified property sheet page.

**FRRibbonBackStageViewItemSetDisable**

Sets the back stage view item to be disabled or not.

**FRRibbonBackStageViewItemSetDlgItemMaxSize**

Sets the max size of the dialog item in the back stage view.

**FRRibbonBackStageViewItemSetGroupTitle**

Sets the title of the property sheet group.

**FRRibbonBackStageViewItemSetTitlePage**

Sets the title of the property sheet page.

**FRRibbonBackStageViewItemSetPropertyActivePage**

Sets the active property sheet page.

**FRRibbonBackStageViewItemSetTitle**

Sets the title of the back stage view item.

**FRRibbonBackStageViewItemSetVisible**

Sets the back stage view item to be visible or not.

## Functions detail

### FRRibbonBackStageViewItemAddAnchor

#### Syntax

```
FS_BOOL FRRibbonBackStageViewItemAddAnchor (
    HWND hParentWnd,
    HWND hWnd,
    FRRibbonBackStageViewItemXMoveType typeMove,
    FRRibbonBackStageViewItemXSizeType typeSize,
    FS_DevicePoint percMove,
    FS_DevicePoint percSize,
```

---

```
void* pParentWnd
);
```

**Description**

This interface is used to adjust the control position of the dialog added on the back stage view.

**Parameter**

<code>hParentWnd</code>	[In] The parent window handle of the dialog.
<code>hWnd</code>	[In] The window handle of the dialog.
<code>typeMove</code>	[In] The input movement type.
<code>typeSize</code>	[In] The input size type.
<code>percMove</code>	[In] Sets it 100,100 as default.
<code>percSize</code>	[In] Sets it 100,100 as default.
<code>pParentWnd</code>	[In] A pointer to the main frame UI parent window. It represents the <i>MFC CWnd*</i> . Sets it to NULL as default. You can get the parent window through <a href="#">FRDocGetUIParentWnd</a> .

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

fr\_barTempl.h: 4455

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

[FRRibbonBackStageViewItemAddDialog](#)

**Syntax**

```
FS_BOOL FRRibbonBackStageViewItemAddDialog (
    FR.RibbonBackStageViewItem item,
    FRRibbonBackstageCreateProc createProc,
    FRRibbonBackstageDestoryProc destroyProc,
    void* pClientData
);
```

**Description**

You can create your own dialog on the back stage view when it is shown. You can either add a dialog or a property sheet on the back stage view.

**Parameter**


---

item	[In] The input back stage view item object.
createProc	[In] The callback function is called to notify the plug-in to create the dialog when the back stage view is shown.
destroyProc	[In] The callback function is called to notify the plug-in to destroy the dialog when the back stage view is closed.
pClientData	[In] The user-supplied data passed to the callback function.

---

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

fr\_barTempl.h: 1103

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonBackStageViewItemAddPropertySheetGroup****Syntax**

```
FS_BOOL FRRibbonBackStageViewItemAddPropertySheetGroup (
    FR\_RibbonBackStageViewItem item,
    FS_LPCSTR lpsName,
    FS_LPCWSTR lpwsGroupTitle,
    FS_INT32 nPos
);
```

**Description**

Adds a new property sheet group.

**Parameter**


---

item	[In] The input back stage view item object.
------	---

---

---

IpsName	[In] The input name of the property sheet group.
---------	--

---

lpwsGroupTitle	[In] The input title of the property sheet group.
----------------	---

---

nPos	[In] The input position of the property sheet group.
------	--

---

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

fr\_barTempl.h: 4404

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonBackStageViewItemAddPropertySheetPage****Syntax**

```
FS_BOOL FRRibbonBackStageViewItemAddPropertySheetPage (
    FR\_RibbonBackStageViewItem item,
    FS_LPCSTR IpsName,
    FS_LPCWSTR lpwsPageTitle,
    FS_DIBitmap pBitmap,
    FS_INT32 nPos,
    FRRibbonBackstageCreateProc createProc,
    FRRibbonBackstageDestoryProc destroyProc,
    void* pClientData
);
```

**Description**

You can create your own property sheet page on the back stage view when it is shown.  
You can either add a dialog or a property sheet on the back stage view.

**Parameter**


---

item	[In] The input back stage view item object.
------	---

---

IpsName	[In] The input name of the property sheet page.
---------	---

---

lpwsPageTitle	[In] The input title of the property sheet page.
---------------	--

---

pBitmap	[In] The input bitmap of the property sheet page.
---------	---

---

---

nPos	[In] The input position of the property sheet page.
createProc	[In] The callback function is called to notify the plug-in to create the dialog when the back stage view is shown.
destroyProc	[In] The callback function is called to notify the plug-in to destroy the dialog when the back stage view is closed.
pClientData	[In] The user-supplied data passed to the callback function.

---

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

fr\_barTempl.h: 1104

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonBackStageViewItemBCGPreTranslateMessage****Syntax**

```
FS_BOOL FRRibbonBackStageViewItemBCGPreTranslateMessage (
    FR.RibbonBackStageViewItem item,
    FS\_LPCSTR lpsPageName,
    FRRibbonBackstageSelectPageProc selectPageProc
);
```

**Description**

Sets the callback function called to notify the plug-in that the specified property page is selected.

**Parameter**


---

item	[In] The input back stage view item object.
lpsPageName	[In] The specified property page of the back stage view item.

---

selectPageProc [In] The callback function is called to notify the plug-in that the specified property page is selected.

**Return**

void.

**Head file reference**

fr\_barTempl.h: 4599

**Related method****Since**[SDK LATEEST VERSION > 7.3.1](#)**FRRibbonBackStageViewItemEnablePropertyPageRemove****Syntax**

```
void FRRibbonBackStageViewItemEnablePropertyPageRemove (
    FR\_RibbonBackStageViewItem item,
    FS\_LPCSTR lpsName,
    FS\_BOOL bEnable
);
```

**Description**

Sets whether the property page can be removed or not.

**Parameter**

item	[In] The input back stage view item object.
lpsName	[In] The specified name of the property sheet page.
bEnable	[In] Whether the property page can be removed or not.

**Return**

void.

**Head file reference**

fr\_barTempl.h: 4549

**Related method****Since**[SDK LATEEST VERSION > 7.2.2](#)**FRRibbonBackStageViewItemGetElement****Syntax**

```
FR_RibbonElement FRRibbonBackStageViewItemGetElement (
    FR\_RibbonBackStageViewItem item
);
```

**Description**

Gets the ribbon element associated with the back stage view item. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**Parameter**

---

item	[In] The input back stage view item object.
------	---

---

**Return**

The ribbon element associated with the back stage view item.

**Head file reference**

fr\_barTempl.h: 4537

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRRibbonBackStageViewItemIsDialogExist****Syntax**

```
FS_BOOL FRRibbonBackStageViewItemIsDialogExist (
    FR.RibbonBackStageViewItem item
);
```

**Description**

Checks whether the dialog exists or not.

**Parameter**

---

item	[In] The input back stage view item object.
------	---

---

**Return**

[TRUE](#) means the dialog exists, otherwise not.

**Head file reference**

fr\_barTempl.h: 4526

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRRibbonBackStageViewItemIsPropertySheetPageExist**

**Syntax**

```
FS_BOOL FRRibbonBackStageViewItemIsPropertySheetPageExist (
    FR\_RibbonBackStageViewItem item,
    FS\_LPCSTR lpsPageName
);
```

**Description**

Checks whether the property sheet page exists or not.

**Parameter**

item	[In] The input back stage view item object.
lpsPageName	[In] The specified name of the property sheet page.

**Return**

[TRUE](#) means the property sheet page exists, otherwise not.

**Head file reference**

fr\_barTempl.h: 4514

**Related method**

[FRRibbonBackStageViewItemAddPropertySheetPage](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

## FRRibbonBackStageViewItemModifiedToRibbonStyleButton

**Syntax**

```
void* FRRibbonBackStageViewItemModifiedToRibbonStyleButton (
    void* pWnd,
    unsigned int nID,
    FRRibbonStyleButtonType typeButton,
    void* pParentWnd
);
```

**Description**

Modifies the MFC controls to ribbon style button.

**Parameter**

pWnd	[In] The pointer to the MFC control.
nID	[In] The control ID.

---

typeButton	[In] The specified ribbon style type.
pParentWnd	[In] A pointer to the main frame UI parent window. It represents the <i>MFC CWnd*</i> . Sets it to NULL as default. You can get the parent window through <a href="#">FRDocGetUIParentWnd</a> .

---

**Return**

The ribbon style button. For example, if sets *typeButton* as [FR\\_RibbonStyle\\_Button](#) , the returned value can be converted to [FR\\_RibbonStyleButton](#) .

**Head file reference**

fr\_barTempl.h: 4473

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonBackStageViewItemRedrawPropertySheet****Syntax**

```
void FRRibbonBackStageViewItemRedrawPropertySheet (
    FR\_RibbonBackStageViewItem item
);
```

**Description**

Redraws the property sheet.

**Parameter**


---

item	[In] The input back stage view item object.
------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 4562

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRRibbonBackStageViewItemRemovePropertySheetPage****Syntax**

```
FS_BOOL FRRibbonBackStageViewItemRemovePropertySheetPage (
    FR\_RibbonBackStageViewItem item,
```



---

```
FS_LPCSTR lpsName  
);
```

**Description**

Removes the specified property sheet page.

**Parameter**


---

item	[In] The input back stage view item object.
lpsName	[In] The specified name of the property sheet page.

---

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

fr\_barTempl.h: 4392

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonBackStageViewItemSetDisable****Syntax**

```
void FRRibbonBackStageViewItemSetDisable (  
    FR\_RibbonBackStageViewItem item,  
    FS_BOOL bDisable  
)
```

**Description**

Sets the back stage view item to be disabled or not.

**Parameter**


---

item	[In] The input back stage view item object.
bDisable	[In] <a href="#">TRUE</a> if the back stage view item is disabled, otherwise not.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4443



**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonBackStageViewItemSetDlgItemMaxSize****Syntax**

```
void FRRibbonBackStageViewItemSetDlgItemMaxSize (
    HWND hParentWnd,
    HWND hWnd,
    FS\_INT32 cx,
    FS\_INT32 cy
);
```

**Description**

Sets the max size of the dialog item in the back stage view.

**Parameter**

<a href="#">hParentWnd</a>	[In] The parent window of the dialog item.
----------------------------	--

<a href="#">hWnd</a>	[In] The window handle of the dialog item.
----------------------	--

<a href="#">cx</a>	[In] The horizon size.
--------------------	------------------------

<a href="#">cy</a>	[In] The vertical size.
--------------------	-------------------------

**Return**

void.

**Head file reference**

[fr\\_barTempl.h: 4573](#)

**Related method****Since**[SDK LATEEST VERSION > 7.2.2](#)**FRRibbonBackStageViewItemSetGroupTitle****Syntax**

```
FS_BOOL FRRibbonBackStageViewItemSetGroupTitle (
    FR\_RibbonBackStageViewItem item,
    FS\_LPCWSTR lpwsGroupTitle,
    int nGroupIndex
);
```



**Description**

Sets the title of the property sheet group.

**Parameter**

item	[In] The input back stage view item object.
lpwsGroupTitle	[In] The input title of the property sheet group.
nGroupIndex	[In] The specified index.

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

fr\_barTempl.h: 4418

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonBackStageViewItemSetTitle****Syntax**

```
FS_BOOL FRRibbonBackStageViewItemSetTitle (
    FR\_RibbonBackStageViewItem item,
    FS\_LPCSTR lpsPageName,
    FS\_LPCWSTR lpwsTitle
);
```

**Description**

Sets the title of the property sheet page.

**Parameter**

item	[In] The input back stage view item object.
lpsPageName	[In] The specified name of the property sheet page.
lpwsTitle	[In] The input title you want to set.

**Return**

[TRUE](#) for success, otherwise not.

#### Head file reference

fr\_barTempl.h: 4501

#### Related method

[FRRibbonBackStageViewItemAddPropertySheetPage](#)

#### Since

[SDK LATEEST VERSION > 2.1.0.1](#)

## FRRibbonBackStageViewItemSetPropertyActivePage

#### Syntax

```
void FRRibbonBackStageViewItemSetPropertyActivePage (
    FR\_RibbonBackStageViewItem item,
    FS\_LPCSTR lpsPageName
);
```

#### Description

Sets the active property sheet page.

#### Parameter

item	[In] The input back stage view item object.
------	---

lpsPageName	[In] The specified name of the property sheet page.
-------------	---

#### Return

void

#### Head file reference

fr\_barTempl.h: 4489

#### Related method

#### Since

[SDK LATEEST VERSION > 2.0](#)

## FRRibbonBackStageViewItemSetTitle

#### Syntax

```
void FRRibbonBackStageViewItemSetTitle (
    FR\_RibbonBackStageViewItem item,
    FS\_LPCWSTR lpwsTitle
);
```

**Description**

Sets the title of the back stage view item.

**Parameter**

---

item	[In] The input back stage view item object.
lpwsTitle	[In] The input title of the back stage view item.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4345

**Related method**

[FRRibbonBarAddBackStageViewItem](#)

[FRRibbonBarGetBackStageViewItem](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonBackStageViewItemSetVisible****Syntax**

```
void FRRibbonBackStageViewItemSetVisible (
    FR\_RibbonBackStageViewItem item,
    FS\_BOOL bVisible
);
```

**Description**

Sets the back stage view item to be visible or not.

**Parameter**

---

item	[In] The input back stage view item object.
bVisible	[In] <a href="#">TRUE</a> if the back stage view item is visible, otherwise not.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4431

**Related method**

Since  
[SDK LATEEST VERSION > 2.0](#)

## FR\_RibbonBar

[Return from Used by](#)

### Description

The Foxit Reader has a ribbon bar in ribbon mode. The ribbon contains several categories. You can get the ribbon bar object by calling [FRAppGetRibbonBar](#) .

### Returned from

[FRAppGetRibbonBar](#)  
[FRAppGetRibbonBar2](#)

### Used by

[FRRibbonBarActivateContextCategory](#)  
[FRRibbonBarAddAsQAT](#)  
[FRRibbonBarAddBackStageViewItem](#)  
[FRRibbonBarAddButtonToAddPlace](#)  
[FRRibbonBarAddCaptionButton](#)  
[FRRibbonBarAddCategory](#)  
[FRRibbonBarAddCategory2](#)  
[FRRibbonBarAddHidden](#)  
[FRRibbonBarAddRibbonContextCategory](#)  
[FRRibbonBarAddToTabs](#)  
[FRRibbonBarCloseFilePage](#)  
[FRRibbonBarFindElementByName](#)  
[FRRibbonBarGetActiveCategory](#)  
[FRRibbonBarGetBackStageViewItem](#)  
[FRRibbonBarGetCategoryByIndex](#)  
[FRRibbonBarGetCategoryByName](#)  
[FRRibbonBarGetCategoryCount](#)  
[FRRibbonBarIsBackStageViewActive](#)  
[FRRibbonBarIsMinimize](#)  
[FRRibbonBarRecalcLayout](#)  
[FRRibbonBarSelectBackStageViewItem](#)  
[FRRibbonBarSetActiveCategory](#)  
[FRRibbonBarSetActiveCategory2](#)  
[FRRibbonBarShowButtonInAddPlace](#)  
[FRRibbonBarShowContextCategories](#)  
[FRRibbonBarUpdateCmdUI](#)

### Callbacks

#### Callbacks summary

[FROnRemovePropertyPage](#)



A callback for ribbon file page event handler.

#### [FROnClickAddPlacePageButton](#)

A callback for ribbon file page event handler.

#### [FROnSelectPropertyPage](#)

A callback for ribbon file page event handler.

## Callbacks detail

### [FROnRemovePropertyPage](#)

#### **Syntax**

```
typedef void (*FROnRemovePropertyPage)(
    FS_LPVVOID clientData,
    FS_LPCSTR lpsName
);
```

#### **Description**

A callback for ribbon file page event handler. It is called whenever a property page is removed.

#### **Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

lpsName	[In] The specified property page name.
---------	--

#### **Return**

#### **Head file reference**

fr\_barExpT.h: 335

#### **Group**

#### [FR\\_RibbonFilePageEventCallbacksRec](#)

#### **Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

### [FROnClickAddPlacePageButton](#)

#### **Syntax**

```
typedef void (*FROnClickAddPlacePageButton)(
    FS_LPVVOID clientData,
    FS_LPCSTR lpsButtonName,
    FS_BOOL bSaveAsItemPage
);
```

#### **Description**

A callback for ribbon file page event handler. It is called whenever a button on a add-a-place property page is clicked.

**Parameter**


---

clientData	[In] The user-supplied data.
lpsButtonName	[In] The specified button name.
bSaveAsItemPage	[In] Indicates whether the property page is under save as item or not.

---

**Return****Head file reference**

fr\_barExpT.h: 350

**Group**[FR\\_RibbonFilePageEventCallbacksRec](#)**Since**[SDK\\_LATEEST\\_VERSION > 8.0.2](#)**FROnSelectPropertyPage****Syntax**

```
typedef void (*FROnSelectPropertyPage)(
    FS_LPVOID clientData,
    FS_LPCSTR lpsElementName,
    FS_LPCSTR lpsPageName
);
```

**Description**

A callback for ribbon file page event handler. It is called whenever a property page is selected.

**Parameter**


---

clientData	[In] The user-supplied data.
lpsElementName	[In] The specified ribbon element name.
lpsPageName	[In] The specified ribbon property page name.

---

**Return****Head file reference**

fr\_barExpT.h: 365



**Group**[FR\\_RibbonFilePageEventCallbacksRec](#)**Since**[SDK LATEEST VERSION > 8.0.2](#)

## Functions

### Functions summary

[FRRibbonBarActivateContextCategory](#)

Activates the specified context category.

[FRRibbonBarAddAsQAT](#)

Adds the specified ribbon button to the quick access toolbar. The quick access toolbar is located at the left-top corner.

[FRRibbonBarAddBackStageViewItem](#)

Adds a new back stage view item under FILE category. The back stage view item is associated with a view. You can add your own dialog or property sheet page on the view.

[FRRibbonBarAddButtonToAddPlace](#)

There are two back stage view items under FILE category, *Open* and *Save As*. You can add a place under these two back stage view items. So the user can open file from other places such as the file server.

[FRRibbonBarAddCaptionButton](#)

Adds a ribbon button to the caption bar on the right.

[FRRibbonBarAddCategory](#)

Adds a new category to the ribbon bar. A category object can be used to manage the operation categories. For example, all the commenting tools are included in the COMMENT category.

[FRRibbonBarAddCategory2](#)

Adds a new category to the ribbon bar. A category object can be used to manage the operation categories. For example, all the commenting tools are included in the COMMENT category.

[FRRibbonBarAddHidden](#)

Adds a hidden ribbon button. It can be added to QAT by [FRRibbonBarAddAsQAT](#).

[FRRibbonBarAddRibbonContextCategory](#)

Adds a new context category to the ribbon bar.

[FRRibbonBarAddToTabs](#)

Adds a new ribbon button. The ribbon button will be added to the right-top corner of the ribbon bar.

[FRRibbonBarCloseFilePage](#)

Closes the page that is shown by clicking the FILE category.

[FRRibbonBarDestroyFilePageEventHandler](#)

Destroys the ribbon file page event handler returned by [FRRibbonBarRegisterFilePageEventHandler](#).

[FRRibbonBarFindElementByName](#)

Gets the specified element.

[FRRibbonBarGetActiveCategory](#)

Gets the active category object.

[FRRibbonBarGetBackStageViewItem](#)

Gets the specified back stage view item object.



**FRRibbonBarGetCategoryByIndex**

Gets the specified ribbon category object.

**FRRibbonBarGetCategoryByName**

Gets the specified ribbon category object.

**FRRibbonBarGetCategoryCount**

Gets the count of categories.

**FRRibbonBarIsBackStageViewActive**

Checks whether the back view is active or not.

**FRRibbonBarIsMinimize**

Checks whether the ribbon bar is minimized or not.

**FRRibbonBarRecalcLayout**

Whether to recalc the layout of the panels or not.

**FRRibbonBarRegisterFilePageEventHandler**

Registers a callbacks set for ribbon file page event handler.

**FRRibbonBarSelectBackStageViewItem**

Selects the specified back stage view item object.

**FRRibbonBarSetActiveCategory**

Sets the active category.

**FRRibbonBarSetActiveCategory2**

Sets the active category.

**FRRibbonBarShowButtonInAddPlace**

There are two back stage view items under FILE category, *Open* and *Save As*. You can add a place under these two back stage view items. So the user can open file from other places such as the file server. You can show or hide the buttons in the Add-a-Place.

**FRRibbonBarShowContextCategories**

Shows or hides the specified context category.

**FRRibbonBarUpdateCmdUI**

Updates the UI state of ribbon button.

## Functions detail

### FRRibbonBarActivateContextCategory

#### Syntax

```
FS_BOOL FRRibbonBarActivateContextCategory (
    FR.RibbonBar ribbonBar,
    unsigned int uiContextID
);
```

#### Description

Activates the specified context category.

#### Parameter

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

uiContextID	[In] The specified ID of context category.
-------------	--

#### Return

TRUE for success, otherwise failure.



**Head file reference**

fr\_barTempl.h: 987

**Related method**[FRRibbonBarShowContextCategories](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonBarAddAsQAT****Syntax**

```
void FRRibbonBarAddAsQAT (
    FR.RibbonBar ribbonBar,
    unsigned int uiBtnID,
    BOOL bVisible
);
```

**Description**

Adds the specified ribbon button to the quick access toolbar. The quick access toolbar is located at the left-top corner.

**Parameter**

---

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

---

uiBtnID	[In] The specified ID of ribbon button.
---------	---

---

bVisible	[In] Whether the ribbon button on the quick access toolbar is visible or not.
----------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1052

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonBarAddBackStageViewItem****Syntax**

```
FR.RibbonBackStageViewItem FRRibbonBarAddBackStageViewItem (
```



---

```
FR.RibbonBar ribbonBar,
const FS_CHAR* name,
FS_LPCWSTR lpwsTitle,
FS_INT32 nPos
);
```

**Description**

Adds a new back stage view item under FILE category. The back stage view item is associated with a view. You can add your own dialog or property sheet page on the view.

**Parameter**

ribbonBar	[In] The input ribbon bar.
name	[In] The input name of the back stage view item.
lpwsTitle	[In] The input title of the back stage view item.
nPos	[In] The input position of the back stage view item.

**Return**

The new back stage view item object.

**Head file reference**

fr\_barTempl.h: 1093

**Related method**

[FRRibbonBackStageViewItemAddDialog](#)

[FRRibbonBackStageViewItemAddPropertySheetPage](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonBarAddButtonToAddPlace****Syntax**

```
FS_BOOL FRRibbonBarAddButtonToAddPlace (
    FR.RibbonBar ribbonBar,
    const FS_CHAR* name,
    FS_LPCWSTR lpwsTitle,
    FRExecuteProc proc,
    FS_DIBitmap pImage,
    FS_BOOL bToSaveAsItem,
    void* pClientData
);
```

**Description**

There are two back stage view items under FILE category, *Open* and *Save As*. You can add a place under these two back stage view items. So the user can open file from other places such as the file server.

### Parameter

---

ribbonBar	[In] The input ribbon bar.
name	[In] The name of the button.
lpwsTitle	[In] The title of the button.
proc	[In] This callback will be called when the user clicks the button.
pImage	[In] The input bitmap of the button
bToSaveAsItem	[In] Whether the button need to be added to <i>Save As</i> item under FILE category.
pClientData	[In] The user-supplied data.

---

### Return

[TRUE](#) for success, otherwise failure.

### Head file reference

fr\_barTempl.h: 1135

### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonBarAddCaptionButton

### Syntax

```
FR_RibbonButton FRRibbonBarAddCaptionButton (
    FR_RibbonBar ribbonBar,
    const FS_CHAR* name,
    FS_LPCWSTR lpwsTitle,
    FS_INT32 nIndex
);
```

### Description

Adds a ribbon button to the caption bar on the right.



**Parameter**


---

ribbonBar	[In] The input ribbon bar.
name	[In] The input name of the button.
lpwsTitle	[In] The input title of the button.
nIndex	[In] The input position of the button. Sets it -1 as default.

---

**Return**

The ribbon button added to the caption bar on the right.

**Head file reference**

fr\_barTempl.h: 1250

**Related method****Since**

[SDK LATEEST VERSION > 7.2.2](#)

**FRRibbonBarAddCategory****Syntax**

```
FR_RibbonCategory FRRibbonBarAddCategory (
    FR_RibbonBar ribbonBar,
    const FS_CHAR* name,
    FS_LPCWSTR lpwsTitle
);
```

**Description**

Adds a new category to the ribbon bar. A category object can be used to manage the operation categories. For example, all the commenting tools are included in the COMMENT category.

**Parameter**


---

ribbonBar	[In] The input ribbon bar.
name	[In] The input name of the category.
lpwsTitle	[In] The input title of the category.

---

**Return**

The category added to the ribbon bar.



**Head file reference**

fr\_barTempl.h: 961

**Related method**[FRAppGetRibbonBar](#)**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FRRibbonBarAddCategory2****Syntax**

```
FR_RibbonCategory FRRibbonBarAddCategory2 (
    FR\_RibbonBar ribbonBar,
    const FS\_CHAR\* name,
    FS\_LPCWSTR lpwsTitle,
    FS\_INT32 nPos
);
```

**Description**

Adds a new category to the ribbon bar. A category object can be used to manage the operation categories. For example, all the commenting tools are included in the COMMENT category.

**Parameter**

---

ribbonBar	[In] The input ribbon bar.
name	[In] The input name of the category.
lpwsTitle	[In] The input title of the category.
nPos	[In] The position of the category in the ribbon bar.

---

**Return**

The category added to the ribbon bar.

**Head file reference**

fr\_barTempl.h: 1286

**Related method**[FRAppGetRibbonBar](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRRibbonBarAddHidden****Syntax**

```
FR_RibbonButton FRRibbonBarAddHidden (
    FR.RibbonBar ribbonBar,
    const FS_CHAR* name,
    FS_LPCWSTR lpwsTitle
);
```

**Description**

Adds a hidden ribbon button. It can be added to QAT by [FRRibbonBarAddAsQAT](#).

**Parameter**

ribbonBar	[In] The input ribbon bar.
name	[In] The input name of the ribbon button.
lpwsTitle	[In] The input title of the ribbon button.

**Return**

The new ribbon button.

**Head file reference**

fr\_barTempl.h: 1080

**Related method****Since**

[SDK LATEST VERSION > 2.0](#)

**FRRibbonBarAddRibbonContextCategory****Syntax**

```
FR_RibbonCategory FRRibbonBarAddRibbonContextCategory (
    FR.RibbonBar ribbonBar,
    unsigned int uiContextID,
    const FS_CHAR* categoryName,
    FS_LPCWSTR lpwsCategoryTitle,
    FS_LPCWSTR lpwsContextTitle,
    FS_LPCWSTR lpwsShortcutKey
);
```

**Description**

Adds a new context category to the ribbon bar.

**Parameter**

---

ribbonBar	[In] The input ribbon bar.
uiContextID	[In] The context category ID.
categoryName	[In] The name of the category.
lpwsCategoryTitle	[In] The title of the category.
lpwsContextTitle	[In] The title of the context category.
lpwsShorcutKey	[In] The shortcut key of the context category.

---

**Return**

The context category added to the ribbon bar.

**Head file reference**

fr\_barTempl.h: 975

**Related method**

[FRRibbonBarAddCategory](#)

[FRRibbonBarActivateContextCategory](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonBarAddToTabs****Syntax**

```
void* FRRibbonBarAddToTabs (
    FR\_RibbonBar ribbonBar,
    FR\_Ribbon\_Element\_Type nElementType,
    const FS\_CHAR\* name,
    FS\_LPCWSTR lpwsTitle,
    FS\_INT32 nPos
);
```

**Description**

Adds a new ribbon button. The ribbon button will be added to the right-top corner of the ribbon bar.

**Parameter**


---

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

---



---

nElementType	[In] The input type of the ribbon button.
name	[In] The input name of the ribbon button.
lpwsTitle	[In] The input title of the ribbon button.
nPos	[In] The input position of the ribbon button.

---

**Return**

A new ribbon button.

**Head file reference**

fr\_barTempl.h: 1065

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonBarCloseFilePage****Syntax**

```
void FRRibbonBarCloseFilePage (
    FR.RibbonBar ribbonBar
);
```

**Description**

Closes the page that is shown by clicking the FILE category.

**Parameter**


---

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1153

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonBarDestroyFilePageEventHandler**

**Syntax**

```
void FRRibbonBarDestroyFilePageEventHandler (
    void* eventHandler
);
```

**Description**

Destroys the ribbon file page event handler returned by [FRRibbonBarRegisterFilePageEventHandler](#) .

**Parameter**

---

eventHandler	[In] The pointer to ribbon file page event handler returned by <a href="#">FRRibbonBarRegisterFilePageEventHandler</a> .
--------------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 1267

**Related method**

[FRRibbonBarRegisterFilePageEventHandler](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRRibbonBarFindElementByName****Syntax**

```
FR_RibbonElement FRRibbonBarFindElementByName (
    FR_RibbonBar ribbonBar,
    const FS_CHAR* categoryName,
    const FS_CHAR* panelName,
    const FS_CHAR* btnName
);
```

**Description**

Gets the specified element.

**Parameter**

---

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

---

categoryName	[In] The specified name of the category.
--------------	--

---

panelName	[In] The specified name of the panel.
-----------	---------------------------------------

---

---

btnName	[In] The specified name of the element.
---------	---

---

**Return**

The specified element.

**Head file reference**

fr\_barTempl.h: 1164

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonBarGetActiveCategory****Syntax**

```
FR_RibbonCategory FRRibbonBarGetActiveCategory (
    FR.RibbonBar ribbonBar
);
```

**Description**

Gets the active category object.

**Parameter**

---

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

---

**Return**

The active category object.

**Head file reference**

fr\_barTempl.h: 1201

**Related method**

[FRRibbonBarAddCategory](#)

[FRRibbonBarSetActiveCategory](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonBarGetBackStageViewItem****Syntax**

```
FR_RibbonBackStageViewItem FRRibbonBarGetBackStageViewItem (
    FR.RibbonBar ribbonBar,
    const FS\_CHAR* name
);
```



**Description**

Gets the specified back stage view item object.

**Parameter**

---

ribbonBar	[In] The input ribbon bar.
name	[In] The specified name of back stage view item.

---

**Return**

The specified back stage view item object.

**Head file reference**

fr\_barTempl.h: 1109

**Related method**

[FRRibbonBarAddBackStageViewItem](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonBarGetCategoryByIndex

**Syntax**

```
FR_RibbonCategory FRRibbonBarGetCategoryByIndex (
    FR.RibbonBar ribbonBar,
    FS\_INT32 nIndex
);
```

**Description**

Gets the specified ribbon category object.

**Parameter**

---

ribbonBar	[In] The input ribbon bar.
nIndex	[In] The specified index.

---

**Return**

The specified ribbon category object.

**Head file reference**

fr\_barTempl.h: 1028

**Related method**

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonBarGetCategoryByName****Syntax**

```
FR_RibbonCategory FRRibbonBarGetCategoryByName (
    FR_RibbonBar ribbonBar,
    const FS_CHAR* categoryName
);
```

**Description**

Gets the specified ribbon category object.

**Parameter**

---

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

---

categoryName	[In] The specified name of the the category.
--------------	--

---

**Return**

The specified ribbon category object.

**Head file reference**

fr\_barTempl.h: 1040

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonBarGetCategoryCount****Syntax**

```
FS_INT32 FRRibbonBarGetCategoryCount (
    FR_RibbonBar ribbonBar
);
```

**Description**

Gets the count of categories.

**Parameter**

---

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

---

**Return**

The count of categories.

**Head file reference**

fr\_barTempl.h: 1017

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonBarIsBackStageViewActive****Syntax**

```
FS_BOOL FRRibbonBarIsBackStageViewActive (
    FR.RibbonBar ribbonBar
);
```

**Description**

Checks whether the back view is active or not.

**Parameter**

---

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

---

**Return**

[TRUE](#) if the back view is active, otherwise not.

**Head file reference**

fr\_barTempl.h: 1130

**Related method**

[FRRibbonBarSelectBackStageViewItem](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonBarIsMinimize****Syntax**

```
FS_BOOL FRRibbonBarIsMinimize (
    FR.RibbonBar ribbonBar
);
```

**Description**

Checks whether the ribbon bar is minimized or not.

**Parameter**

---

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

---

**Return**

[TRUE](#) means the ribbon bar is minimized, otherwise not.

**Head file reference**

fr\_barTempl.h: 1239

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRRibbonBarRecalcLayout****Syntax**

```
void FRRibbonBarRecalcLayout (
    FR.RibbonBar ribbonBar,
    FS\_BOOL bRecalcPanels
);
```

**Description**

Whether to recalc the layout of the panels or not.

**Parameter**


---

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

---

bRecalcPanels	[In] Whether to recalc the layout of the panels.
---------------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1189

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonBarRegisterFilePageEventHandler****Syntax**

```
void* FRRibbonBarRegisterFilePageEventHandler (
    FR.RibbonFilePageEventCallbacks callbacks
);
```



**Description**

Registers a callbacks set for ribbon file page event handler.

**Parameter**

---

callbacks	[In] The input callbacks for ribbon file page event handler.
-----------	--

---

**Return**

The pointer to ribbon file page event handler can be destroyed by [FRRibbonBarDestroyFilePageEventHandler](#).

**Head file reference**

fr\_barTempl.h: 1264

**Related method**

[FRRibbonBarDestroyFilePageEventHandler](#)

**Since**

[SDK LATEEST VERSION > 7.2.2](#)

**FRRibbonBarSelectBackStageViewItem****Syntax**

```
void FRRibbonBarSelectBackStageViewItem (
    FR.RibbonBar ribbonBar,
    const FS_CHAR* name
);
```

**Description**

Selects the specified back stage view item object.

**Parameter**

---

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

---

---

name	[In] The specified name of back stage view item.
------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1121

**Related method**

[FRRibbonBarAddBackStageViewItem](#)



---

[FRRibbonBarGetBackStageViewItem](#)  
[FRRibbonBarIsBackStageViewActive](#)

**Since**  
[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonBarSetActiveCategory

### Syntax

```
FS_BOOL FRRibbonBarSetActiveCategory (
    FR.RibbonBar ribbonBar,
    const FS\_CHAR\* categoryName
);
```

### Description

Sets the active category.

### Parameter

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

categoryName	[In] The specified name of the category.
--------------	--

### Return

[TRUE](#) for success, otherwise failure.

### Head file reference

fr\_barTempl.h: 1208

### Related method

[FRRibbonBarAddCategory](#)  
[FRRibbonBarGetActiveCategory](#)  
[FRRibbonBarSetActiveCategory2](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonBarSetActiveCategory2

### Syntax

```
FS_BOOL FRRibbonBarSetActiveCategory2 (
    FR.RibbonBar ribbonBar,
    FR.RibbonCategory ribbonCategory
);
```

### Description

Sets the active category.



**Parameter**


---

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

---

ribbonCategory	[In] The specified category object.
----------------	-------------------------------------

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_barTempl.h: 1222

**Related method**

[FRRibbonBarSetActiveCategory](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonBarShowButtonInAddPlace****Syntax**

```
FS_BOOL FRRibbonBarShowButtonInAddPlace (
    FR_RibbonBar ribbonBar,
    const FS_CHAR* name,
    FS_BOOL bShow,
    FS_BOOL bToSaveAsItem
);
```

**Description**

There are two back stage view items under FILE category, *Open* and *Save As*. You can add a place under these two back stage view items. So the user can open file from other places such as the file server. You can show or hide the buttons in the Add-a-Place.

**Parameter**


---

ribbonBar	[In] The input ribbon bar.
-----------	----------------------------

---

name	[In] The name of the button.
------	------------------------------

---

bShow	[In] T Whether to show the button or not.
-------	---

---

bToSaveAsItem	[In] Whether the button is under <i>Save As</i> item under FILE category.
---------------	---

---

**Return**

[TRUE](#) for success, otherwise failure.

#### Head file reference

fr\_barTempl.h: 1301

#### Related method

[FRRibbonBarAddButtonToAddPlace](#)

#### Since

[SDK LATEEST VERSION > 8.1](#)

## FRRibbonBarShowContextCategories

#### Syntax

```
void FRRibbonBarShowContextCategories (
    FR.RibbonBar ribbonBar,
    unsigned int uiContextID,
    FS.Bool bShow
);
```

#### Description

Shows or hides the specified context category.

#### Parameter

---

ribbonBar	[In] The input ribbon bar.
uiContextID	[In] The specified ID of context category.
bShow	[In] Whether to show or hide the context category.

---

#### Return

void

#### Head file reference

fr\_barTempl.h: 999

#### Related method

[FRRibbonBarAddCategory](#)

#### Since

[SDK LATEEST VERSION > 2.0](#)

## FRRibbonBarUpdateCmdUI

#### Syntax

```
void FRRibbonBarUpdateCmdUI (
```



```
FR_RibbonBar ribbonBar,  
const FS_CHAR* name  
);
```

**Description**

Updates the UI state of ribbon button.

**Parameter**

---

ribbonBar	[In] The input ribbon bar.
name	[In] The name of the button.

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 1317

**Since**

[SDK LATEEST VERSION > 9.1](#)

## FR\_RibbonButton

**[Return from Used by](#)****Description**

The ribbon button is a type of ribbon element.

**Returned from**

[FRRibbonBarAddCaptionButton](#)  
[FRRibbonBarAddHidden](#)  
[FRRibbonPanelSetLaunchButton](#)

**Used by**

[FRRibbonButtonCloseButtonPopupWnd](#)  
[FRRibbonButtonGetElement](#)  
[FRRibbonButtonSetAlwaysShowDescription](#)  
[FRRibbonButtonSetButtonPopupWnd](#)  
[FRRibbonButtonSetDefaultCommand](#)

**Functions****Functions summary**

**FRRibbonButtonCloseButtonPopupWnd**

Closes the pop-up window created by [FRRibbonButtonSetButtonPopupWnd](#)

**FRRibbonButtonGetElement**

Gets the ribbon element associated with the ribbon button. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**FRRibbonButtonSetAlwaysShowDescription**

Whether to show the description always like the title or not.

**FRRibbonButtonSetButtonPopupWnd**

If a ribbon button has the drop-down style, you can pop up a window when it is dropped down. Sets *hWnd* as NULL to make the ribbon button has the drop-down arrow. When the [FRDropDownProc](#) callback is invoked, calls this interface in the callback function to pass the window handle to the ribbon button.

**FRRibbonButtonSetDefaultCommand**

When the button has the drop-down style, sets *bSet* as [TRUE](#) if you wan to both click the button to execute and drop down the button, sets *bSet* as [FALSE](#) if you just want to drop down the button.

## Functions detail

### FRRibbonButtonCloseButtonPopupWnd

**Description**

Closes the pop-up window created by [FRRibbonButtonSetButtonPopupWnd](#)

**Parameter**

ribbonButton	[In] The input ribbon button object.
--------------	--------------------------------------

**Return**

void

**Head file reference**

fr\_barTempl.h: 2579

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

### FRRibbonButtonGetElement

**Syntax**

```
FR_RibbonElement FRRibbonButtonGetElement (
    FR_RibbonButton ribbonButton
);
```

**Description**

Gets the ribbon element associated with the ribbon button. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**Parameter**

---

ribbonButton	[In] The input ribbon button object.
--------------	--------------------------------------

---

**Return**

The ribbon element associated with the ribbon button.

**Head file reference**

fr\_barTempl.h: 2527

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonButtonSetAlwaysShowDescription****Syntax**

```
void FRRibbonButtonSetAlwaysShowDescription (
    FR.RibbonButton ribbonButton,
    FS\_BOOL bAlwaysShowDescription
);
```

**Description**

Whether to show the description always like the title or not.

**Parameter**

---

ribbonButton	[In] The input ribbon button object.
--------------	--------------------------------------

---

bAlwaysShowDescription [In] [TRUE](#) if you want to show the description always like the title.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2553

**Related method**

**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonButtonSetButtonPopupWnd****Syntax**

```
void FRRibbonButtonSetButtonPopupWnd (
    FR.RibbonButton ribbonButton,
    HWND hWnd
);
```

**Description**

If a ribbon button has the drop-down style, you can pop up a window when it is dropped down. Sets *hWnd* as NULL to make the ribbon button has the drop-down arrow. When the [FRDropDownProc](#) callback is invoked, calls this interface in the callback function to pass the window handle to the ribbon button.

**Parameter**

ribbonButton	[In] The input ribbon button object.
--------------	--------------------------------------

hWnd	[In] The window handle.
------	-------------------------

**Return**

void

**Head file reference**

fr\_barTempl.h: 2565

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonButtonSetDefaultCommand****Syntax**

```
void FRRibbonButtonSetDefaultCommand (
    FR.RibbonButton ribbonButton,
    FS\_BOOL bSet
);
```

**Description**

When the button has the drop-down style, sets *bSet* as [TRUE](#) if you wan to both click the button to execute and drop down the button, sets *bSet* as [FALSE](#) if you just want to drop down the button.

**Parameter**

---

ribbonButton	[In] The input ribbon button object.
bSet	[In] When the button has the drop-down style, sets it as <a href="#">TRUE</a> if you wan to both click the button to execute and drop down the button, sets it as <a href="#">FALSE</a> if you just want to drop down the button.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2539

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)

## FR\_RibbonCategory

[Return from](#) [Used by](#)

**Description**

A [FR\\_RibbonCategory](#) object can be used to manage the operation categories. For example, all the commenting tools re included in the COMMENT category. For further classification, you have to add ribbon panels to the ribbon categories.

**Returned from**

[FRRibbonBarAddCategory](#)  
[FRRibbonBarAddCategory2](#)  
[FRRibbonBarAddRibbonContextCategory](#)  
[FRRibbonBarGetActiveCategory](#)  
[FRRibbonBarGetCategoryByIndex](#)  
[FRRibbonBarGetCategoryByName](#)

**Used by**

[FRRibbonBarSetActiveCategory2](#)  
[FRRibbonCategoryAddDialog](#)  
[FRRibbonCategoryAddDialogToRight](#)  
[FRRibbonCategoryAddPanel](#)  
[FRRibbonCategoryAddPanel2](#)  
[FRRibbonCategoryCopyPanel](#)  
[FRRibbonCategoryGetContextTitle](#)  
[FRRibbonCategoryGetHighlightColor](#)  
[FRRibbonCategoryGetKey](#)  
[FRRibbonCategoryGetName](#)  
[FRRibbonCategoryGetPanelByIndex](#)  
[FRRibbonCategoryGetPanelByName](#)

---

[FRRibbonCategoryGetPanelCount](#)  
[FRRibbonCategoryGetPos](#)  
[FRRibbonCategoryGetTitle](#)  
[FRRibbonCategoryGetVisible](#)  
[FRRibbonCategoryPreTranslateMessage](#)  
[FRRibbonCategoryRecalcLayout](#)  
[FRRibbonCategorySetContextTitle](#)  
[FRRibbonCategorySetHighlight](#)  
[FRRibbonCategorySetKey](#)  
[FRRibbonCategorySetPos](#)  
[FRRibbonCategorySetTitle](#)  
[FRRibbonCategorySetVisible](#)  
[FRRibbonCategoryShowInQATCustomizeToolsDlg](#)

## Callbacks

### Callbacks summary

#### [FRCategoryDlgCreateProc](#)

The callback function is called to notify the plug-in to create the dialog attached to the category.

#### [FRCategoryDlgDestoryProc](#)

The callback function is called to notify the plug-in to destroy the dialog attached to the category.

### Callbacks detail

#### FRCategoryDlgCreateProc

##### Syntax

```
typedef(HWND (*FRCategoryDlgCreateProc)(  
    HWND hParentWnd,  
    void* pDialog  
)
```

##### Description

The callback function is called to notify the plug-in to create the dialog attached to the category.

##### Parameter

---

hParentWnd	[In] The parent window handle.
------------	--------------------------------

---

pDialog	[In] The pointer to a <i>MFC CDialog</i> attached to the category. It is passed from <a href="#"><u>FRRibbonCategoryAddDialog</u></a> .
---------	---

---

##### Return

The dialog handle created.

##### Head file reference

fr\_barExpT.h: 431

**Related method**[FRRibbonCategoryAddDialog](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)**FRCategoryDlgDestoryProc****Syntax**

```
typedef void (*FRCategoryDlgDestoryProc)(  
    void* pDialog  
)
```

**Description**

The callback function is called to notify the plug-in to destroy the dialog attached to the category.

**Parameter**

---

pDialog	[In] The dialog handle to be destroyed. It is passed from <a href="#"><u>FRRibbonCategoryAddDialog</u></a> .
---------	--

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 443

**Related method**[FRRibbonCategoryAddDialog](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)

## Functions

### Functions summary

**[FRRibbonCategoryAddDialog](#)**

You can create your own dialog under the category. You have to invoke [FRRibbonCategoryPreTranslateMessage](#) in the dialog's *PreTranslateMessage* function to dispatch the message.

**[FRRibbonCategoryAddDialogToRight](#)**

You can create your own dialog under the category on the right. You have to invoke [FRRibbonCategoryPreTranslateMessage](#) in the dialog's *PreTranslateMessage* function to dispatch the message.

**[FRRibbonCategoryAddPanel](#)**

Adds a new ribbon panel to the ribbon category. A ribbon category may contain several ribbon panels.

**FRRibbonCategoryAddPanel2**

Adds a new ribbon panel to the ribbon category. A ribbon category may contain several ribbon panels.

**FRRibbonCategoryCopyPanel**

Gets a copy of the specified ribbon panel object.

**FRRibbonCategoryGetContextTitle**

Gets the title of the ribbon context category.

**FRRibbonCategoryGetHighlightColor**

Gets the color when the category is highlight.

**FRRibbonCategoryGetKey**

Gets the shortcut key of the ribbon category.

**FRRibbonCategoryGetName**

Gets the name of the ribbon category.

**FRRibbonCategoryGetPanelByIndex**

Gets the ribbon panel by index.

**FRRibbonCategoryGetPanelByName**

Gets the ribbon panel by name.

**FRRibbonCategoryGetPanelCount**

Gets the count of ribbon panels.

**FRRibbonCategoryGetPos**

Gets the position of the ribbon category in the ribbon bar.

**FRRibbonCategoryGetTitle**

Gets the title of the ribbon category.

**FRRibbonCategoryGetVisible**

Checks whether the ribbon category is visible or not.

**FRRibbonCategoryPreTranslateMessage**

When you create a dialog under the category, you have to invoke

[FRRibbonCategoryPreTranslateMessage](#) in the dialog's *PreTranslateMessage* function to dispatch the message.

**FRRibbonCategoryRecalcLayout**

Whether to recalc the layout of the panels or not.

**FRRibbonCategorySetContextTitle**

Sets the title of the ribbon context category.

**FRRibbonCategorySetHighlight**

Sets the category to be highlight. Invokes [FRRibbonCategoryRecalcLayout](#) to take it effect.

**FRRibbonCategorySetKey**

Sets the shortcut key of the ribbon category.

**FRRibbonCategorySetPos**

Sets the position of the ribbon category in the ribbon bar.

**FRRibbonCategorySetTitle**

Sets the title of the ribbon category.

**FRRibbonCategorySetVisible**

Sets the ribbon category to visible or invisible.

**FRRibbonCategoryShowInQATCustomizeToolsDlg**

Sets whether the category can be shown in the QAT customize tools dialog.

## Functions detail

**FRRibbonCategoryAddDialog****Syntax**

```
FS_BOOL FRRibbonCategoryAddDialog (
    FR.RibbonCategory ribbonCategory,
    FRCategoryDlgCreateProc createProc,
    FRCategoryDlgDestoryProc destroyProc,
    void* pDialog
);
```

**Description**

You can create your own dialog under the category. You have to invoke [FRRibbonCategoryPreTranslateMessage](#) in the dialog's *PreTranslateMessage* function to dispatch the message.

**Parameter**

ribbonCategory	[In] The input ribbon category object.
createProc	[In] The callback function is called to notify the plug-in to create the dialog attached to the category.
destroyProc	[In] The callback function is called to notify the plug-in to destroy the dialog attached to the category.
pDialog	[In] The input pointer to a <i>MFC CDialog</i> attached to the category. It will be passed to the <i>createProc</i> and <i>destroyProc</i> .

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

[fr\\_barTempl.h](#): 1546

**Related method****Since**

[SDK LATEEST VERSION > 7.1](#)

**FRRibbonCategoryAddDialogToRight****Syntax**

```
FS_BOOL FRRibbonCategoryAddDialogToRight (
    FR.RibbonCategory ribbonCategory,
    FRCategoryDlgCreateProc createProc,
    FRCategoryDlgDestoryProc destroyProc,
    void* pDialog
);
```

**Description**

You can create your own dialog under the category on the right. You have to invoke [FRRibbonCategoryPreTranslateMessage](#) in the dialog's *PreTranslateMessage* function to dispatch the message.

**Parameter**


---

ribbonCategory	[In] The input ribbon category object.
createProc	[In] The callback function is called to notify the plug-in to create the dialog attached to the category.
destroyProc	[In] The callback function is called to notify the plug-in to destroy the dialog attached to the category.
pDialog	[In] The input pointer to a <i>MFC CDialog</i> attached to the category. It will be passed to the <i>createProc</i> and <i>destroyProc</i> .

---

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

fr\_barTempl.h: 1574

**Related method**

[FRRibbonCategoryAddDialog](#)

**Since**

[SDK LATEEST VERSION > 7.3.1](#)

**FRRibbonCategoryAddPanel****Syntax**

```
FR_RibbonPanel FRRibbonCategoryAddPanel (
    FR.RibbonCategory ribbonCategory,
    const FS\_CHAR\* name,
    FS\_LPCWSTR lpwsTitle,
    FS\_DIBitmap defBtnBitmap
);
```

**Description**

Adds a new ribbon panel to the ribbon category. A ribbon category may contain several ribbon panels.

**Parameter**


---

ribbonCategory	[In] The input ribbon category object.
----------------	--

---

---

name	[In] The input name of the ribbon panel.
------	--

---

lpwsTitle	[In] The input title of the ribbon panel.
-----------	---

---

defBtnBitmap	[In] The default icon of the ribbon panel.
--------------	--

---

**Return**

The new ribbon panel added to the ribbon category.

**Head file reference**

fr\_barTempl.h: 1431

**Related method**

[FRRibbonCategoryGetPanelCount](#)

[FRRibbonCategoryGetPanelByIndex](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonCategoryAddPanel2****Syntax**

```
FR_RibbonPanel FRRibbonCategoryAddPanel2 (
    FR\_RibbonCategory ribbonCategory,
    const FS\_CHAR\* name,
    FS\_LPCWSTR lpwsTitle,
    FS\_DIBitmap defBtnBitmap,
    FS\_INT32 nPos
);
```

**Description**

Adds a new ribbon panel to the ribbon category. A ribbon category may contain several ribbon panels.

**Parameter**


---

ribbonCategory	[In] The input ribbon category object.
----------------	--

---

name	[In] The input name of the ribbon panel.
------	--

---

lpwsTitle	[In] The input title of the ribbon panel.
-----------	---

---

defBtnBitmap	[In] The default icon of the ribbon panel.
--------------	--

---



---

nPos	[In] The position of the ribbon panel.
------	--

---

**Return**

The new ribbon panel added to the ribbon category.

**Head file reference**

fr\_barTempl.h: 1529

**Related method**

[FRRibbonCategoryGetPanelCount](#)  
[FRRibbonCategoryGetPanelByIndex](#)  
[FRRibbonCategoryAddPanel](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 3.0.0.0](#)

**FRRibbonCategoryCopyPanel****Syntax**

```
FR_RibbonPanel FRRibbonCategoryCopyPanel (
    FR.RibbonCategory ribbonCategory,
    FR.RibbonPanel ribbonPanel
);
```

**Description**

Gets a copy of the specified ribbon panel object.

**Parameter**


---

ribbonCategory	[In] The input ribbon category object.
----------------	--

---



---

ribbonPanel	[In] The specified ribbon panel object to be copied.
-------------	--

---

**Return**

A copy of the specified ribbon panel object.

**Head file reference**

fr\_barTempl.h: 1505

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRRibbonCategoryGetContextTitle****Syntax**

```
void FRRibbonCategoryGetContextTitle (
    FR.RibbonCategory ribbonCategory,
    FS.WideString* wsTitle
);
```

**Description**

Gets the title of the ribbon context category.

**Parameter**

ribbonCategory	[In] The input ribbon category object.
wsTitle	[Out] It receives the title of the ribbon context category.

**Return**

void

**Head file reference**

fr\_barTempl.h: 1372

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonCategoryGetHighlightColor****Syntax**

```
FS_COLORREF FRRibbonCategoryGetHighlightColor (
    FR.RibbonCategory ribbonCategory
);
```

**Description**

Gets the color when the category is highlight.

**Parameter**

ribbonCategory	[In] The input ribbon category object.
----------------	--

**Return**

The color when the category is highlight.

**Head file reference**

fr\_barTempl.h: 1620

**Related method**

[FRRibbonCategorySetHighlight](#)



**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

## FRRibbonCategoryGetKey

**Syntax**

```
void FRRibbonCategoryGetKey (
    FR.RibbonCategory ribbonCategory,
    FS.WideString* wsKey
);
```

**Description**

Gets the shortcut key of the ribbon category.

**Parameter**

---

ribbonCategory	[In] The input ribbon category object.
----------------	--

---

wsKey	[Out] It receives the shortcut key of the ribbon category.
-------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1396

**Related method**

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonCategoryGetName

**Syntax**

```
void FRRibbonCategoryGetName (
    FR.RibbonCategory ribbonCategory,
    FS.ByteString* bsName
);
```

**Description**

Gets the name of the ribbon category.

**Parameter**

---

ribbonCategory	[In] The input ribbon category object.
----------------	--

---



---

bsName	[Out] It receives the name of the ribbon category.
--------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1481

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonCategoryGetPanelByIndex****Syntax**

```
FR_RibbonPanel FRRibbonCategoryGetPanelByIndex (
    FR.RibbonCategory ribbonCategory,
    FS\_INT32 nIndex
);
```

**Description**

Gets the ribbon panel by index.

**Parameter**


---

ribbonCategory	[In] The input ribbon category object.
----------------	--

---

nIndex	[In] The specified index.
--------	---------------------------

---

**Return**

The specified ribbon panel.

**Head file reference**

fr\_barTempl.h: 1441

**Related method**[FRRibbonCategoryGetPanelByName](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonCategoryGetPanelByName****Syntax**

```
FR_RibbonPanel FRRibbonCategoryGetPanelByName (
    FR.RibbonCategory ribbonCategory,
```



```
const FS_CHAR* panelName  
);
```

**Description**

Gets the ribbon panel by name.

**Parameter**

---

ribbonCategory	[In] The input ribbon category object.
----------------	--

---

panelName	[In] The specified name.
-----------	--------------------------

---

**Return**

The specified ribbon panel.

**Head file reference**

fr\_barTempl.h: 1464

**Related method**

[FRRibbonCategoryGetPanelByIndex](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonCategoryGetPanelCount

**Syntax**

```
FS_INT32 FRRibbonCategoryGetPanelCount (  
    FR.RibbonCategory ribbonCategory  
);
```

**Description**

Gets the count of ribbon panels.

**Parameter**

---

ribbonCategory	[In] The input ribbon category object.
----------------	--

---

**Return**

The count of ribbon panels

**Head file reference**

fr\_barTempl.h: 1440

**Related method****Since**

## [SDK\\_LATEEST\\_VERSION > 2.0](#)

### FRRibbonCategoryGetPos

#### Syntax

```
FS_INT32 FRRibbonCategoryGetPos (
    FR.RibbonCategory ribbonCategory
);
```

#### Description

Gets the position of the ribbon category in the ribbon bar.

#### Parameter

---

ribbonCategory	[In] The input ribbon category object.
----------------	--

---

#### Return

The position of the ribbon category in the ribbon bar.

#### Head file reference

fr\_barTempl.h: 1589

#### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

### FRRibbonCategoryGetTitle

#### Syntax

```
void FRRibbonCategoryGetTitle (
    FR.RibbonCategory ribbonCategory,
    FS WideString* wsTitle
);
```

#### Description

Gets the title of the ribbon category.

#### Parameter

---

ribbonCategory	[In] The input ribbon category object.
----------------	--

---

wsTitle	[Out] It receives the title of the ribbon category.
---------	---

---

#### Return

void

**Head file reference**

fr\_barTempl.h: 1348

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonCategoryGetVisible****Syntax**

```
FS_BOOL FRRibbonCategoryGetVisible (
    FR.RibbonCategory ribbonCategory
);
```

**Description**

Checks whether the ribbon category is visible or not.

**Parameter**

---

ribbonCategory	[In] The input ribbon category object.
----------------	--

---

**Return**

[TRUE](#) if the ribbon category is visible, otherwise invisible.

**Head file reference**

fr\_barTempl.h: 1420

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonCategoryPreTranslateMessage****Syntax**

```
FS_BOOL FRRibbonCategoryPreTranslateMessage (
    FR.RibbonCategory ribbonCategory,
    void* pMsg
);
```

**Description**

When you create a dialog under the category, you have to invoke [FRRibbonCategoryPreTranslateMessage](#) in the dialog's *PreTranslateMessage* function to dispatch the message.

**Parameter**

---

ribbonCategory	[In] The input ribbon category object.
pMsg	[In] Pointer to a <i>MFC MSG</i> structure that contains the message to process.

---

**Return**

[TRUE](#) if the message was fully processed and should not be processed further. [FALSE](#) if the message should be processed in the normal way.

**Head file reference**

fr\_barTempl.h: 1547

**Related method**

[FRRibbonCategoryAddDialog](#)

**Since**

[SDK LATEEST VERSION > 7.1](#)

**FRRibbonCategoryRecalcLayout****Syntax**

```
void FRRibbonCategoryRecalcLayout (
    FR\_RibbonCategory ribbonCategory,
    FS\_BOOL bRecalcPanels
);
```

**Description**

Whether to recalc the layout of the panels or not.

**Parameter**


---

ribbonCategory	[In] The input ribbon category object.
bRecalcPanels	[In] Whether to recalc the layout of the panels.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1517

**Related method****Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FRRibbonCategorySetContextTitle****Syntax**

```
void FRRibbonCategorySetContextTitle (
    FR.RibbonCategory ribbonCategory,
    FS\_LPCWSTR lpwsTitle
);
```

**Description**

Sets the title of the ribbon context category.

**Parameter**

ribbonCategory	[In] The input ribbon category object.
----------------	--

lpwsTitle	[In] The input title of the ribbon context category.
-----------	--

**Return**

void

**Head file reference**

[fr\\_barTempl.h](#): 1360

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonCategorySetHighlight****Syntax**

```
void FRRibbonCategorySetHighlight (
    FR.RibbonCategory ribbonCategory,
    FS\_COLORREF color,
    FS\_BOOL bOnceOnly
);
```

**Description**

Sets the category to be highlight. Invokes [FRRibbonCategoryRecalcLayout](#) to take it effect.

**Parameter**

ribbonCategory	[In] The input ribbon category object.
----------------	--

color	[In] The color when the category is highlight.
-------	--



---

bOnceOnly	[In] If TRUE, the highlight will disappear when the category is activated.
-----------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 1612

**Related method**[FRRibbonCategoryGetHighlightColor](#)**Since**[SDK LATEEST VERSION > 7.3.1](#)**FRRibbonCategorySetKey****Syntax**

```
void FRRibbonCategorySetKey (
    FR\_RibbonCategory ribbonCategory,
    FS\_LPCWSTR lpwsShorcutKey
);
```

**Description**

Sets the shortcut key of the ribbon category.

**Parameter**


---

ribbonCategory	[In] The input ribbon category object.
----------------	--

---

lpwsShorcutKey	[In] The input shortcut key of the ribbon category.
----------------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1384

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonCategorySetPos****Syntax**

```
void FRRibbonCategorySetPos (
```

```
FR.RibbonCategory ribbonCategory,  
FS_INT32 nPos  
);
```

**Description**

Sets the position of the ribbon category in the ribbon bar.

**Parameter**

---

ribbonCategory	[In] The input ribbon category object.
----------------	--

---

nPos	[In] The position of the ribbon category in the ribbon bar.
------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 1600

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRRibbonCategorySetTitle****Syntax**

```
void FRRibbonCategorySetTitle (  
    FR.RibbonCategory ribbonCategory,  
    FS_LPCWSTR lpwsTitle  
) ;
```

**Description**

Sets the title of the ribbon category.

**Parameter**

---

ribbonCategory	[In] The input ribbon category object.
----------------	--

---

lpwsTitle	[In] The input title of the ribbon category.
-----------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1336

**Related method**[FRRibbonBarAddCategory](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonCategorySetVisible****Syntax**

```
void FRRibbonCategorySetVisible (
    FR.RibbonCategory ribbonCategory,
    FS\_BOOL bVisible
);
```

**Description**

Sets the ribbon category to visible or invisible.

**Parameter**

ribbonCategory	[In] The input ribbon category object.
bVisible	[In] <a href="#">TRUE</a> if the ribbon category is visible, otherwise is invisible.

**Return**

void

**Head file reference**

fr\_barTempl.h: 1408

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonCategoryShowInQATCustomizeToolsDlg****Syntax**

```
void FRRibbonCategoryShowInQATCustomizeToolsDlg (
    FR.RibbonCategory ribbonCategory,
    FS\_BOOL bShow
);
```

**Description**

Sets whether the category can be shown in the QAT customize tools dialog.

**Parameter**


---

ribbonCategory	[In] The input ribbon category object.
bShow	[In] It indicates whether the category can be shown in the QAT customize tools dialog.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1493

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)

## FR\_RibbonCheckBox

**Used by****Description**

The ribbon check box is a type of ribbon element.

**Used by**

[\*\*FRRibbonCheckBoxGetElement\*\*](#)  
[\*\*FRRibbonCheckBoxIsChecked\*\*](#)  
[\*\*FRRibbonCheckBoxSetCheck\*\*](#)

**Functions****Functions summary****FRRibbonCheckBoxGetElement**

Gets the ribbon element associated with the ribbon check box. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**FRRibbonCheckBoxIsChecked**

Checks whether the ribbon check box is checked or not.

**FRRibbonCheckBoxSetCheck**

Whether to set the ribbon check box to be checked or not.

**Functions detail****FRRibbonCheckBoxGetElement**

**Syntax**

```
FR_RibbonElement FRRibbonCheckBoxGetElement (
    FR\_RibbonCheckBox ribbonCheckBox
);
```

**Description**

Gets the ribbon element associated with the ribbon check box. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**Parameter**

---

ribbonCheckBox	[In] The input ribbon check box object.
----------------	---

---

**Return**

The ribbon element associated with the ribbon check box.

**Head file reference**

fr\_barTempl.h: 2735

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonCheckBoxIsChecked****Syntax**

```
FS_BOOL FRRibbonCheckBoxIsChecked (
    FR\_RibbonCheckBox ribbonCheckBox
);
```

**Description**

Checks whether the ribbon check box is checked or not.

**Parameter**

---

ribbonCheckBox	[In] The input ribbon check box object.
----------------	---

---

**Return**

[TRUE](#) if the ribbon check box is checked, otherwise not.

**Head file reference**

fr\_barTempl.h: 2747

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonCheckBoxSetCheck****Syntax**

```
void FRRibbonCheckBoxSetCheck (
    FR\_RibbonCheckBox ribbonCheckBox,
    FS\_BOOL bCheck
);
```

**Description**

Whether to set the ribbon check box to be checked or not.

**Parameter**

ribbonCheckBox	[In] The input ribbon check box object.
bCheck	[In] <a href="#">TRUE</a> if you want to set the ribbon check box to be checked.

**Return**

void

**Head file reference**

fr\_barTempl.h: 2758

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FR\_RibbonColorButton

**Used by****Description**

The ribbon color button is a type of ribbon element.

**Used by**

[FRRibbonColorButtonAddAdditionalColor](#)  
[FRRibbonColorButtonAddColorsGroup](#)  
[FRRibbonColorButtonAddMainColor](#)  
[FRRibbonColorButtonAddStandardColor](#)  
[FRRibbonColorButtonEnableAutomaticButton](#)  
[FRRibbonColorButtonEnableOtherButton](#)

[\*\*FRRibbonColorButtonGetAutomaticButtonLabel\*\*](#)  
[\*\*FRRibbonColorButtonGetAutomaticButtonToolTip\*\*](#)  
[\*\*FRRibbonColorButtonGetColor\*\*](#)  
[\*\*FRRibbonColorButtonGetElement\*\*](#)  
[\*\*FRRibbonColorButtonGetGroupLabel\*\*](#)  
[\*\*FRRibbonColorButtonGetOtherButtonLabel\*\*](#)  
[\*\*FRRibbonColorButtonGetOtherButtonToolTip\*\*](#)  
[\*\*FRRibbonColorButtonIsAutomaticButtonClick\*\*](#)  
[\*\*FRRibbonColorButtonSetAutomaticButtonLabel\*\*](#)  
[\*\*FRRibbonColorButtonSetAutomaticButtonToolTip\*\*](#)  
[\*\*FRRibbonColorButtonSetColor\*\*](#)  
[\*\*FRRibbonColorButtonSetColorBoxSize\*\*](#)  
[\*\*FRRibbonColorButtonSetColumns\*\*](#)  
[\*\*FRRibbonColorButtonSetDefaultCommand\*\*](#)  
[\*\*FRRibbonColorButtonSetGroupLabel\*\*](#)  
[\*\*FRRibbonColorButtonSetOtherButtonLabel\*\*](#)  
[\*\*FRRibbonColorButtonSetOtherButtonToolTip\*\*](#)

## Functions

### Functions summary

[\*\*FRRibbonColorButtonAddAdditionalColor\*\*](#)  
Adds the group of the additional colors.

[\*\*FRRibbonColorButtonAddColorsGroup\*\*](#)  
Adds a new group of colors.

[\*\*FRRibbonColorButtonAddMainColor\*\*](#)  
Adds the group of the main colors.

[\*\*FRRibbonColorButtonAddStandardColor\*\*](#)  
Adds the group of the standard colors.

[\*\*FRRibbonColorButtonEnableAutomaticButton\*\*](#)  
Enables the automatic button or not.

[\*\*FRRibbonColorButtonEnableOtherButton\*\*](#)  
Enables the button to select other color.

[\*\*FRRibbonColorButtonGetAutomaticButtonLabel\*\*](#)  
Gets the label of the automatic button.

[\*\*FRRibbonColorButtonGetAutomaticButtonToolTip\*\*](#)  
Gets the tooltip of the automatic button.

[\*\*FRRibbonColorButtonGetColor\*\*](#)  
Gets the selected color of the ribbon color button.

[\*\*FRRibbonColorButtonGetElement\*\*](#)  
Gets the ribbon element associated with the ribbon color button. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

[\*\*FRRibbonColorButtonGetGroupLabel\*\*](#)  
Gets the label of the group.

[\*\*FRRibbonColorButtonGetOtherButtonLabel\*\*](#)  
Gets the label of the button use to select other color.

[\*\*FRRibbonColorButtonGetOtherButtonToolTip\*\*](#)  
Gets the tooltip of the button use to select other color.

[\*\*FRRibbonColorButtonIsAutomaticButtonClick\*\*](#)  
Checks whether the automatic button is clicked or not.

**FRRibbonColorButtonSetAutomaticButtonLabel**

Sets the label of the automatic button.

**FRRibbonColorButtonSetAutomaticButtonToolTip**

Sets the tooltip of the automatic button.

**FRRibbonColorButtonSetColor**

Sets the selected color of the ribbon color button.

**FRRibbonColorButtonSetColorBoxSize**

Sets the size of the color box.

**FRRibbonColorButtonSetColumns**

Sets the column numbers.

**FRRibbonColorButtonSetDefaultCommand**

When the button has the drop-down style, sets *bDefaultCommand* as **TRUE** if you wan to both click the button to execute and drop down the button, sets *bDefaultCommand* as **FALSE** if you just want to drop down the button.

**FRRibbonColorButtonSetGroupLabel**

Sets the group label.

**FRRibbonColorButtonSetOtherButtonLabel**

Sets the label of the button used to select other color.

**FRRibbonColorButtonSetOtherButtonToolTip**

Sets the tooltip of the button used to select other color.

## Functions detail

### FRRibbonColorButtonAddAdditionalColor

#### Syntax

```
void FRRibbonColorButtonAddAdditionalColor (
    FR_RibbonColorButton ribbonColorButton,
    const FS_CHAR* name,
    FS_LPCWSTR lpwsLabel,
    FS_BOOL bContiguousColumns
);
```

#### Description

Adds the group of the additional colors.

#### Parameter

---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

name	[In] The name of the group.
------	-----------------------------

---

lpwsLabel	[In] The label of the group.
-----------	------------------------------

---

bContiguousColumns	[In] Sets TRUE if the columns are continuous.
--------------------	---

---

#### Return

void

**Head file reference**

fr\_barTempl.h: 3868

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonColorButtonAddColorsGroup****Syntax**

```
void FRRibbonColorButtonAddColorsGroup (
    FR\_RibbonColorButton ribbonColorButton,
    const FS\_CHAR\* name,
    FS\_LPCWSTR lpwsLabel,
    FS\_COLORREF\* colors,
    FS\_INT32 nColorCount,
    FS\_BOOL bContiguousColumns
);
```

**Description**

Adds a new group of colors.

**Parameter**

---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

name	[In] The name of the group.
------	-----------------------------

---

lpwsLabel	[In] The label of the group.
-----------	------------------------------

---

colors	[In] The specified colors of the group.
--------	---

---

nColorCount	[In] The count of the colors.
-------------	-------------------------------

---

bContiguousColumns	[In] Sets TRUE if the columns are continuous.
--------------------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3838

**Related method**

**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonColorButtonAddMainColor****Syntax**

```
void FRRibbonColorButtonAddMainColor (
    FR.RibbonColorButton ribbonColorButton,
    const FS\_CHAR\* name,
    FS\_LPCWSTR lpwsLabel,
    FS\_BOOL bContiguousColumns
);
```

**Description**

Adds the group of the main colors.

**Parameter**

---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

name	[In] The name of the group.
------	-----------------------------

---

lpwsLabel	[In] The label of the group.
-----------	------------------------------

---

bContiguousColumns	[In] Sets TRUE if the columns are continuous.
--------------------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3882

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonColorButtonAddStandardColor****Syntax**

```
void FRRibbonColorButtonAddStandardColor (
    FR.RibbonColorButton ribbonColorButton,
    const FS\_CHAR\* name,
    FS\_LPCWSTR lpwsLabel,
    FS\_BOOL bContiguousColumns
);
```

**Description**

Adds the group of the standard colors.

**Parameter**

---

ribbonColorButton	[In] The input ribbon color button object.
name	[In] The name of the group.
lpwsLabel	[In] The label of the group.
bContiguousColumns	[In] Sets TRUE if the columns are continuous.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3854

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonColorButtonEnableAutomaticButton****Syntax**

```
void FRRibbonColorButtonEnableAutomaticButton (
    FR\_RibbonColorButton ribbonColorButton,
    FS\_LPCWSTR lpwsLabel,
    FS\_COLORREF colorAutomatic,
    FS\_BOOL bEnable,
    FS\_LPCWSTR lpwsToolTip,
    FS\_BOOL bOnTop,
    FS\_BOOL bDrawBorder
);
```

**Description**

Enables the automatic button or not.

**Parameter**

---

ribbonColorButton	[In] The input ribbon color button object.
lpwsLabel	[In] The input label of the automatic button.

---

---

colorAutomatic	[In] The input automatic color.
bEnable	[In] Sets TRUE to enable the automatic button.
lpwsToolTip	[In] The input tooltip of the automatic button.
bOnTop	[In] Sets TRUE to set the automatic button on the top.
bDrawBorder	[In] Sets TRUE to draw the border.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3712

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonColorButtonEnableOtherButton****Syntax**

```
void FRRibbonColorButtonEnableOtherButton (
    FR.RibbonColorButton ribbonColorButton,
    FS\_LPCWSTR lpwsLabel,
    FS\_LPCWSTR lpwsToolTip
);
```

**Description**

Enables the button to select other color.

**Parameter**


---

ribbonColorButton	[In] The input ribbon color button object.
lpwsLabel	[In] The input label.
lpwsToolTip	[In] The input tooltip.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3753

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

## FRRibbonColorButtonGetAutomaticButtonLabel

#### Syntax

```
void FRRibbonColorButtonGetAutomaticButtonLabel (
    FR.RibbonColorButton ribbonColorButton,
    FS.WideString* wsLabel
);
```

#### Description

Gets the label of the automatic button.

#### Parameter

---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

wsLabel	[Out] It receives the label of the automatic button.
---------	--

---

#### Return

void

#### Head file reference

fr\_barTempl.h: 3790

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

## FRRibbonColorButtonGetAutomaticButtonToolTip

#### Syntax

```
void FRRibbonColorButtonGetAutomaticButtonToolTip (
    FR.RibbonColorButton ribbonColorButton,
    FS.WideString* wsTooltip
);
```

#### Description

Gets the tooltip of the automatic button.

#### Parameter



---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

wsTooltip	[Out] It receives the tooltip of the automatic button.
-----------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3802

**Related method****Since**

[SDK LATEST VERSION > 2.0](#)

**FRRibbonColorButtonGetColor****Syntax**

```
FS_COLORREF FRRibbonColorButtonGetColor (
    FR.RibbonColorButton ribbonColorButton
);
```

**Description**

Gets the selected color of the ribbon color button.

**Parameter**

---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

**Return**

The color.

**Head file reference**

fr\_barTempl.h: 3934

**Related method****Since**

[SDK LATEST VERSION > 2.0](#)

**FRRibbonColorButtonGetElement****Syntax**

```
FR.RibbonElement FRRibbonColorButtonGetElement (
    FR.RibbonColorButton ribbonColorButton
);
```



**Description**

Gets the ribbon element associated with the ribbon color button. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**Parameter**

---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

**Return**

The ribbon element associated with the ribbon color button.

**Head file reference**

fr\_barTempl.h: 3700

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonColorButtonGetGroupLabel****Syntax**

```
void FRRibbonColorButtonGetGroupLabel (
    FR.RibbonColorButton ribbonColorButton,
    const FS\_CHAR* name,
    FS\_WideString* wsLabel
);
```

**Description**

Gets the label of the group.

**Parameter**

---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

---

name	[In] The name of the group.
------	-----------------------------

---

---

wsLabel	[Out] It receives the label of the group.
---------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3909

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonColorButtonGetOtherButtonLabel****Syntax**

```
void FRRibbonColorButtonGetOtherButtonLabel (
    FR.RibbonColorButton ribbonColorButton,
    FS.WideString* wsLabel
);
```

**Description**

Gets the label of the button use to select other color.

**Parameter**

---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

wsLabel	[Out] It receives the label of the button.
---------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3814

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonColorButtonGetOtherButtonToolTip****Syntax**

```
void FRRibbonColorButtonGetOtherButtonToolTip (
    FR.RibbonColorButton ribbonColorButton,
    FS.WideString* wsTooltip
);
```

**Description**

Gets the tooltip of the button use to select other color.

**Parameter**

---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

wsTooltip	[Out] It receives the tooltip of the button.
-----------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3826

**Related method****Since**[SDK LATEST VERSION > 2.0](#)**FRRibbonColorButtonIsAutomaticButtonClick****Syntax**

```
FS_BOOL FRRibbonColorButtonIsAutomaticButtonClick (
    FR.RibbonColorButton ribbonColorButton
);
```

**Description**

Checks whether the automatic button is clicked or not.

**Parameter**


---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

**Return**[TRUE](#) if the automatic button is clicked, otherwise not.**Head file reference**

fr\_barTempl.h: 3984

**Related method****Since**[SDK LATEST VERSION > 2.0](#)**FRRibbonColorButtonSetAutomaticButtonLabel****Syntax**

```
void FRRibbonColorButtonSetAutomaticButtonLabel (
    FR.RibbonColorButton ribbonColorButton,
    FS.LPCWSTR ipwsLabel
);
```

**Description**

Sets the label of the automatic button.

**Parameter**

---

ribbonColorButton	[In] The input ribbon color button object.
lpwsLabel	[In] The input label of the automatic button.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3729

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonColorButtonSetAutomaticButtonToolTip****Syntax**

```
void FRRibbonColorButtonSetAutomaticButtonToolTip (
    FR.RibbonColorButton ribbonColorButton,
    FS\_LPCWSTR lpwsToolTip
);
```

**Description**

Sets the tooltip of the automatic button.

**Parameter**

---

ribbonColorButton	[In] The input ribbon color button object.
lpwsToolTip	[In] The input tooltip of the automatic button.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3741

**Related method****Since**

## [SDK\\_LATEEST\\_VERSION > 2.0](#)

### FRRibbonColorButtonSetColor

#### Syntax

```
void FRRibbonColorButtonSetColor (
    FR.RibbonColorButton ribbonColorButton,
    FS\_COLORREF color
);
```

#### Description

Sets the selected color of the ribbon color button.

#### Parameter

---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

color	[In] The input color.
-------	-----------------------

---

#### Return

void

#### Head file reference

fr\_barTempl.h: 3922

#### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

### FRRibbonColorButtonSetColorBoxSize

#### Syntax

```
void FRRibbonColorButtonSetColorBoxSize (
    FR.RibbonColorButton ribbonColorButton,
    FS\_INT32 nWidth,
    FS\_INT32 nHeight
);
```

#### Description

Sets the size of the color box.

#### Parameter

---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---



---

nWidth	[In] The input width. The default value is 22.
--------	--

---

nHeight	[In] The input height. The default value is 22.
---------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3957

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonColorButtonSetColumns****Syntax**

```
void FRRibbonColorButtonSetColumns (
    FR.RibbonColorButton ribbonColorButton,
    FS\_INT32 nColumns
);
```

**Description**

Sets the column numbers.

**Parameter**

---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

nColumns	[In] The input column numbers.
----------	--------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3945

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonColorButtonSetDefaultCommand****Syntax**

```
void FRRibbonColorButtonSetDefaultCommand (
```

[FR.RibbonColorButton](#) ribbonColorButton,

[FS\\_BOOL](#) bDefaultCommand

);

### Description

When the button has the drop-down style, sets *bDefaultCommand* as [TRUE](#) if you wan to both click the button to execute and drop down the button, sets *bDefaultCommand* as [FALSE](#) if you just want to drop down the button.

### Parameter

---

ribbonColorButton	[In] The input ribbon color button object.
bDefaultCommand	[In] When the button has the drop-down style, sets it as <a href="#">TRUE</a> if you wan to both click the button to execute and drop down the button, sets it as <a href="#">FALSE</a> if you just want to drop down the button.

---

### Return

void

### Head file reference

fr\_barTempl.h: 3970

### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonColorButtonSetGroupLabel

### Syntax

```
void FRRibbonColorButtonSetGroupLabel (
```

[FR.RibbonColorButton](#) ribbonColorButton,

[const FS\\_CHAR\\*](#) name,

[FS\\_LPCWSTR](#) ipwsLabel

);

### Description

Sets the group label.

### Parameter

---

ribbonColorButton	[In] The input ribbon color button object.
name	[In] The name of the group.

---

---

lpwsLabel	[In] The input label of the group.
-----------	------------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3896

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonColorButtonSetOtherButtonLabel****Syntax**

```
void FRRibbonColorButtonSetOtherButtonLabel (
    FR.RibbonColorButton ribbonColorButton,
    FS\_LPCWSTR lpwsLabel
);
```

**Description**

Sets the label of the button used to select other color.

**Parameter**

---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

---

lpwsLabel	[In] The input label.
-----------	-----------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3766

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonColorButtonSetOtherButtonToolTip****Syntax**

```
void FRRibbonColorButtonSetOtherButtonToolTip (
    FR.RibbonColorButton ribbonColorButton,
    FS\_LPCWSTR lpwsToolTip
);
```

**Description**

Sets the tooltip of the button used to select other color.

**Parameter**

---

ribbonColorButton	[In] The input ribbon color button object.
-------------------	--

---

lpwsToolTip	[In] The input tooltip.
-------------	-------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3778

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FR\_RibbonComboBox

### Used by

#### Description

The ribbon combo box is a type of ribbon element.

#### Used by

[FRRibbonComboBoxAddItem](#)  
[FRRibbonComboBoxDeleteItem](#)  
[FRRibbonComboBoxEnableDropDownListResize](#)  
[FRRibbonComboBoxFindItem](#)  
[FRRibbonComboBoxGetCurSel](#)  
[FRRibbonComboBoxGetEditText](#)  
[FRRibbonComboBoxGetElement](#)  
[FRRibbonComboBoxGetHWnd](#)  
[FRRibbonComboBoxGetItem](#)  
[FRRibbonComboBoxGetItemCount](#)  
[FRRibbonComboBoxInsertItem](#)  
[FRRibbonComboBoxRemoveAllItems](#)  
[FRRibbonComboBoxSelectItem](#)  
[FRRibbonComboBoxSetEditText](#)  
[FRRibbonComboBoxSetFocus](#)  
[FRRibbonComboBoxSetTextFlag](#)

**FRRibbonComboBoxSetWidth**

## Functions

### Functions summary

**FRRibbonComboBoxAddItem**

Adds a new item to the ribbon combo box.

**FRRibbonComboBoxDeleteItem**

Deletes the specified item by index.

**FRRibbonComboBoxEnableDropDownListResize**

Sets whether the dropped down list can be resized or not.

**FRRibbonComboBoxFindItem**

Finds the item by text.

**FRRibbonComboBoxGetCurSel**

Gets the currently selected item index.

**FRRibbonComboBoxGetEditText**

Gets the text of the edit text.

**FRRibbonComboBoxGetElement**

Gets the ribbon element associated with the ribbon combo box. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**FRRibbonComboBoxGetHWnd**

Gets the window handle of the ribbon combo box.

**FRRibbonComboBoxGetItem**

Gets the text of the ribbon combo box by index.

**FRRibbonComboBoxGetItemCount**

Gets the item count of the ribbon combo box.

**FRRibbonComboBoxInsertItem**

Inserts a new item to the ribbon combo box by position.

**FRRibbonComboBoxRemoveAllItems**

Removes all the items.

**FRRibbonComboBoxSelectItem**

Selects the specified items by index.

**FRRibbonComboBoxSetEditBox**

Sets whether the ribbon combo box has the edit box or not.

**FRRibbonComboBoxSetEditText**

Sets the edit text of the ribbon combo box.

**FRRibbonComboBoxSetFocus**

Sets the focus to the ribbon combo box or not.

**FRRibbonComboBoxSetTextFlag**

Sets the type you can input into the combo box. 0 for no restriction, 1 for float, 2 for integer.

**FRRibbonComboBoxSetWidth**

Sets the width of the ribbon combo box.

## Functions detail

**FRRibbonComboBoxAddItem**

### Syntax

```
FS_INT32 FRRibbonComboBoxAddItem (
    FR\_RibbonComboBox ribbonComboBox,
    FS\_LPCWSTR lpwsItemText
);
```

**Description**

Adds a new item to the ribbon combo box.

**Parameter**

---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

lpwsItemText	[In] The input text of the new item.
--------------	--------------------------------------

---

**Return**

The index of the new item.

**Head file reference**

fr\_barTempl.h: 2833

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonComboBoxDeleteItem****Syntax**

```
FS_BOOL FRRibbonComboBoxDeleteItem (
    FR\_RibbonComboBox ribbonComboBox,
    FS\_INT32 nItemIndex
);
```

**Description**

Deletes the specified item by index.

**Parameter**

---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

nItemIndex	[In] The specified index.
------------	---------------------------

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

---

fr\_barTempl.h: 2916

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

### FRRibbonComboBoxEnableDropDownListResize

#### Syntax

```
void FRRibbonComboBoxEnableDropDownListResize (
    FR\_RibbonComboBox ribbonComboBox,
    FS\_BOOL bEnable
);
```

#### Description

Sets whether the dropped down list can be resized or not.

#### Parameter

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

bEnable	[In] <a href="#">TRUE</a> if the dropped down list can be resized.
---------	--

#### Return

void

#### Head file reference

fr\_barTempl.h: 2988

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

### FRRibbonComboBoxFindItem

#### Syntax

```
FS_INT32 FRRibbonComboBoxFindItem (
    FR\_RibbonComboBox ribbonComboBox,
    FS\_LPCWSTR lpwsItemText
);
```

#### Description

Finds the item by text.

#### Parameter

---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

lpwsItemText	[In] The specified text.
--------------	--------------------------

---

**Return**

The index of found item.

**Head file reference**

fr\_barTempl.h: 2928

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonComboBoxGetCurSel****Syntax**

```
FS_INT32 FRRibbonComboBoxGetCurSel (
    FR\_RibbonComboBox ribbonComboBox
);
```

**Description**

Gets the currently selected item index.

**Parameter**

---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

**Return**

The currently selected item index.

**Head file reference**

fr\_barTempl.h: 2882

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonComboBoxGetEditText****Syntax**

```
void FRRibbonComboBoxGetEditText (
    FR\_RibbonComboBox ribbonComboBox,
    FS\_WideString* wsText
);
```

**Description**

Gets the text of the edit text.

**Parameter**

---

ribbonComboBox	[In] The input ribbon combo box.
wsText	[Out] It receives the text of the edit.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2964

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonComboBoxGetElement****Syntax**

```
FR_RibbonElement FRRibbonComboBoxGetElement (
    FR.RibbonComboBox ribbonComboBox
);
```

**Description**

Gets the ribbon element associated with the ribbon combo box. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**Parameter**

---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

**Return**

The ribbon element associated with the ribbon combo box.

**Head file reference**

fr\_barTempl.h: 2821

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonComboBoxGetHWnd

### Syntax

```
HWND FRRibbonComboBoxGetHWnd (
    FR\_RibbonComboBox ribbonComboBox
);
```

### Description

Gets the window handle of the ribbon combo box.

### Parameter

---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

### Return

The window handle of the ribbon combo box.

### Head file reference

fr\_barTempl.h: 3000

### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 3.0](#)

## FRRibbonComboBoxGetItem

### Syntax

```
void FRRibbonComboBoxGetItem (
    FR\_RibbonComboBox ribbonComboBox,
    FS\_INT32 nIndex,
    FS\_WideString* wsText
);
```

### Description

Gets the text of the ribbon combo box by index.

### Parameter

---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

nIndex	[In] The specified index.
--------	---------------------------

---

wsText	[Out] It receives the text of the ribbon combo box.
--------	---

---

### Return

void

**Head file reference**

fr\_barTempl.h: 2858

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonComboBoxGetItemCount****Syntax**

```
FS_INT32 FRRibbonComboBoxGetItemCount (
    FR\_RibbonComboBox ribbonComboBox
);
```

**Description**

Gets the item count of the ribbon combo box.

**Parameter**

---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

**Return**

The item count.

**Head file reference**

fr\_barTempl.h: 2858

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonComboBoxInsertItem****Syntax**

```
FS_INT32 FRRibbonComboBoxInsertItem (
    FR\_RibbonComboBox ribbonComboBox,
    FS\_INT32 nPos,
    FS\_LPCWSTR lpwsItemText
);
```

**Description**

Inserts a new item to the ribbon combo box by position.

**Parameter**

---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

nPos	[In] The specified position.
------	------------------------------

---

lpwsItemText	[In] The input text of the new item.
--------------	--------------------------------------

---

**Return**

The index of the new item.

**Head file reference**

fr\_barTempl.h: 2845

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonComboBoxRemoveAllItems****Syntax**

```
void FRRibbonComboBoxRemoveAllItems (
    FR\_RibbonComboBox ribbonComboBox
);
```

**Description**

Removes all the items.

**Parameter**


---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2893

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonComboBoxSelectItem****Syntax**

```
FS_BOOL FRRibbonComboBoxSelectItem (
```

```
FR.RibbonComboBox ribbonComboBox,  
FS\_INT32 nItemIndex  
);
```

**Description**

Selects the specified items by index.

**Parameter**

---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

nItemIndex	[In] The specified index.
------------	---------------------------

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_barTempl.h: 2904

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonComboBoxSetEditBox****Syntax**

```
void FRRibbonComboBoxSetEditBox (  
    FR.RibbonComboBox ribbonComboBox,  
    FS\_BOOL bHasEditBox  
) ;
```

**Description**

Sets whether the ribbon combo box has the edit box or not.

**Parameter**

---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

bHasEditBox	[In] <a href="#">TRUE</a> if the combo box has the edit box.
-------------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2952

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonComboBoxSetEditText****Syntax**

```
void FRRibbonComboBoxSetEditText (
    FR\_RibbonComboBox ribbonComboBox,
    FS\_LPCWSTR lpwsText
);
```

**Description**

Sets the edit text of the ribbon combo box.

**Parameter**

---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

lpwsText	[In] The input text.
----------	----------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2976

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonComboBoxSetFocus****Syntax**

```
void FRRibbonComboBoxSetFocus (
    FR\_RibbonComboBox ribbonComboBox,
    FS\_BOOL bFocus
);
```

**Description**

Sets the focus to the ribbon combo box or not.

**Parameter**

---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

bFocus	[In] Inputs TRUE to set the focus to the ribbon combo box.
--------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 3023

**Related method****Since**[SDK LATEEST VERSION > 7.3.1.0](#)**FRRibbonComboBoxSetTextFlag****Syntax**

```
void FRRibbonComboBoxSetTextFlag (
    FR\_RibbonComboBox ribbonComboBox,
    FS\_DWORD nTextFlag
);
```

**Description**

Sets the type you can input into the combo box. 0 for no restriction, 1 for float, 2 for integer.

**Parameter**


---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

nTextFlag	[In] The type you can input into the combo box. 0 for no restriction, 1 for float, 2 for integer.
-----------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 3011

**Related method****Since**[SDK LATEEST VERSION > 7.1.0.0](#)**FRRibbonComboBoxSetWidth**

**Syntax**

```
void FRRibbonComboBoxSetWidth (
    FR\_RibbonComboBox ribbonComboBox,
    FS\_INT32 nWidth
);
```

**Description**

Sets the width of the ribbon combo box.

**Parameter**

---

ribbonComboBox	[In] The input ribbon combo box.
----------------	----------------------------------

---

nWidth	[In] The input width of the ribbon combo box.
--------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2940

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FR\_RibbonEdit

### [Used by](#)

#### **Description**

The ribbon edit is a type of ribbon element.

#### **Used by**

[FRRibbonEditEnableSpinButtons](#)  
[FRRibbonEditGetElement](#)  
[FRRibbonEditGetHWnd](#)  
[FRRibbonEditGetText](#)  
[FRRibbonEditSetFocus](#)  
[FRRibbonEditSetSearchMode](#)  
[FRRibbonEditSetText](#)  
[FRRibbonEditSetTextFlag](#)  
[FRRibbonEditSetWidth](#)

#### **Functions**

## Functions summary

### [\*\*FRRibbonEditEnableSpinButtons\*\*](#)

Enables the spin buttons of the ribbon edit.

### [\*\*FRRibbonEditGetElement\*\*](#)

Gets the ribbon element associated with the ribbon edit. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

### [\*\*FRRibbonEditGetHWnd\*\*](#)

Gets the window handle of the ribbon edit object.

### [\*\*FRRibbonEditGetText\*\*](#)

Gets the text of the ribbon edit.

### [\*\*FRRibbonEditSetFocus\*\*](#)

Sets the focus to the ribbon edit or not.

### [\*\*FRRibbonEditSetSearchMode\*\*](#)

Whether to enable the search mode of the ribbon edit.

### [\*\*FRRibbonEditSetText\*\*](#)

Sets the text of the ribbon edit.

### [\*\*FRRibbonEditSetTextFlag\*\*](#)

Sets the type you can input into the ribbon edit. 0 for no restriction, 1 for float, 2 for integer.

### [\*\*FRRibbonEditSetWidth\*\*](#)

Sets the width of the ribbon edit.

## Functions detail

### FRRibbonEditEnableSpinButtons

#### Syntax

```
void FRRibbonEditEnableSpinButtons (
    FR_RibbonEdit ribbonEdit,
    FS_INT32 nMin,
    FS_INT32 nMax
);
```

#### Description

Enables the spin buttons of the ribbon edit.

#### Parameter

ribbonEdit	[In] The input ribbon edit object.
------------	------------------------------------

nMin	[In] The minimum value of the spin button.
------	--

nMax	[In] The maximum value of the spin button.
------	--

#### Return

void

**Head file reference**

fr\_barTempl.h: 2634

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonEditGetElement****Syntax**

```
FR_RibbonElement FRRibbonEditGetElement (
    FR\_RibbonEdit ribbonEdit
);
```

**Description**

Gets the ribbon element associated with the ribbon edit. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**Parameter**

---

ribbonEdit	[In] The input ribbon edit object.
------------	------------------------------------

---

**Return**

The ribbon element associated with the ribbon edit.

**Head file reference**

fr\_barTempl.h: 2598

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonEditGetHWnd****Syntax**

```
HWND FRRibbonEditGetHWnd (
    FR\_RibbonEdit ribbonEdit
);
```

**Description**

Gets the window handle of the ribbon edit object.

**Parameter**

---

ribbonEdit	[In] The input ribbon edit object.
------------	------------------------------------

---

**Return**

The window handle of the ribbon edit object.

**Head file reference**

fr\_barTempl.h: 2672

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 3.0](#)

**FRRibbonEditGetText****Syntax**

```
void FRRibbonEditGetText (
    FR\_RibbonEdit ribbonEdit,
    FS\_WideString* wsText
);
```

**Description**

Gets the text of the ribbon edit.

**Parameter**


---

ribbonEdit	[In] The input ribbon edit object.
------------	------------------------------------

---



---

wsText	[Out] It receives the text of the ribbon edit.
--------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2610

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonEditSetFocus****Syntax**

```
void FRRibbonEditSetFocus (
    FR\_RibbonEdit ribbonEdit,
    FS\_BOOL bFocus
);
```

**Description**

Sets the focus to the ribbon edit or not.

**Parameter**

---

ribbonEdit	[In] The input ribbon edit object.
------------	------------------------------------

---

bFocus	[In] Inputs TRUE to set the focus to the ribbon edit.
--------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 2695

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1.0](#)

**FRRibbonEditSetSearchMode****Syntax**

```
void FRRibbonEditSetSearchMode (
    FR\_RibbonEdit ribbonEdit,
    FS\_LPCWSTR lpwsPrompt,
    FS\_BOOL bEnable
);
```

**Description**

Whether to enable the search mode of the ribbon edit.

**Parameter**

---

ribbonEdit	[In] The input ribbon edit object.
------------	------------------------------------

---

lpwsPrompt	[In] Then text displayed in the ribbon edit.
------------	--

---

bEnable	[In] <a href="#">TRUE</a> to enable the search mode.
---------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2647

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

### FRRibbonEditSetText

#### Syntax

```
void FRRibbonEditSetText (
    FR\_RibbonEdit ribbonEdit,
    FS\_LPCWSTR lpwsText
);
```

#### Description

Sets the text of the ribbon edit.

#### Parameter

---

ribbonEdit	[In] The input ribbon edit object.
------------	------------------------------------

---

lpwsText	[In] The input text.
----------	----------------------

---

#### Return

void

#### Head file reference

fr\_barTempl.h: 2622

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

### FRRibbonEditSetTextFlag

#### Syntax

```
void FRRibbonEditSetTextFlag (
    FR\_RibbonEdit ribbonEdit,
    FS\_DWORD nTextFlag
);
```

#### Description

Sets the type you can input into the ribbon edit. 0 for no restriction, 1 for float, 2 for integer.

**Parameter**

---

ribbonEdit	[In] The input ribbon edit object.
------------	------------------------------------

---

nTextFlag	[In] The type you can input into the ribbon edit. 0 for no restriction, 1 for float, 2 for integer.
-----------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 2683

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

**FRRibbonEditSetWidth****Syntax**

```
void FRRibbonEditSetWidth (
    FR\_RibbonEdit ribbonEdit,
    FS\_INT32 nWidth
);
```

**Description**

Sets the width of the ribbon edit.

**Parameter**

---

ribbonEdit	[In] The input ribbon edit object.
------------	------------------------------------

---

nWidth	[In] The input width of the ribbon edit.
--------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2660

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FR\_RibbonElement

[Return from Used by](#)

### Description

The ribbon element object is a basic object. It manipulates the common properties of all types of ribbon element. The ribbon element object is a basic object. It manipulates the common properties of all types of ribbon element, such as the title, the description, the icon and so on. A specified type of ribbon element is associated with a ribbon element object. About the type of on element, you can reference to [FR\\_Ribbon\\_Element\\_Type](#). You can add a new ribbon element to the ribbon panel through [FRRibbonPanelAddElement](#).

### Returned from

[FRRibbonBackStageViewItemGetElement](#)  
[FRRibbonBarFindElementByName](#)  
[FRRibbonButtonGetElement](#)  
[FRRibbonCheckBoxGetElement](#)  
[FRRibbonColorButtonGetElement](#)  
[FRRibbonComboBoxGetElement](#)  
[FRRibbonEditGetElement](#)  
[FRRibbonFontComboBoxGetElement](#)  
[FRRibbonLabelGetElement](#)  
[FRRibbonListButtonGetElement](#)  
[FRRibbonPaletteButtonGetElement](#)  
[FRRibbonPanelGetElementByIndex](#)  
[FRRibbonPanelGetElementByName](#)  
[FRRibbonRadioButtonGetElement](#)  
[FRRibbonSliderGetElement](#)  
[FRRibbonElementFindSubElementByIndex](#)  
[FRRibbonElementGetSubElementByName](#)

### Used by

[FRRibbonPanelCopyElementToPanel](#)  
[FRRibbonElementAddChangeImage](#)  
[FRRibbonElementAddSubItem](#)  
[FRRibbonElementChangeImage](#)  
[FRRibbonElementCopyElementAsSubItem](#)  
[FRRibbonElementDoExecuteProc](#)  
[FRRibbonElementFindSubElementByIndex](#)  
[FRRibbonElementGetAccel](#)  
[FRRibbonElementGetButtonMapId](#)  
[FRRibbonElementGetCategoryName](#)  
[FRRibbonElementGetClientData](#)  
[FRRibbonElementGetClientRect](#)  
[FRRibbonElementGetCorrespondingButton](#)  
[FRRibbonElementGetDescription](#)  
[FRRibbonElementGetKey](#)  
[FRRibbonElementGetName](#)  
[FRRibbonElementGetOriginalID](#)  
[FRRibbonElementGetPanelName](#)  
[FRRibbonElementGetSubElementByName](#)

[FRRibbonElementGetSubItemCount](#)  
[FRRibbonElementGetTitle](#)  
[FRRibbonElementGetTooltip](#)  
[FRRibbonElementGetType](#)  
[FRRibbonElementGetVisible](#)  
[FRRibbonElementGetWindowRect](#)  
[FRRibbonElementIsBackstageViewTabElementKeepState](#)  
[FRRibbonElementIsEnabled](#)  
[FRRibbonElementIsMarked](#)  
[FRRibbonElementRemove](#)  
[FRRibbonElementSetAccel](#)  
[FRRibbonElementSetAsSubElement](#)  
[FRRibbonElementSetBackstageViewTabElementKeepState](#)  
[FRRibbonElementSetClientData](#)  
[FRRibbonElementSetComputeEnabledProc](#)  
[FRRibbonElementSetComputeMarkedProc](#)  
[FRRibbonElementSetDescription](#)  
[FRRibbonElementSetDropdownProc](#)  
[FRRibbonElementSetExecuteProc](#)  
[FRRibbonElementSetImage](#)  
[FRRibbonElementSetImageInitProc](#)  
[FRRibbonElementSetImplicitLargeBitmap](#)  
[FRRibbonElementSetKey](#)  
[FRRibbonElementSetSelectOnly](#)  
[FRRibbonElementSetShowMode](#)  
[FRRibbonElementSetTitle](#)  
[FRRibbonElementSetTooltip](#)  
[FRRibbonElementSetTooltipImage](#)  
[FRRibbonElementSetTooltipImageII](#)  
[FRRibbonElementSetVisible](#)  
[FRRibbonElementShowInQATCustomizeToolsDlg](#)

## Enumerations

### Enumerations summary

#### [FR\\_Ribbon\\_Element\\_Type](#)

The type of ribbon element.

### Enumerations detail

#### [FR\\_Ribbon\\_Element\\_Type](#)

##### Syntax

```
enum FR_Ribbon_Element_Type{  
    FR_RIBBON_NULL,  
    FR_RIBBON_BUTTON,  
    FR_RIBBON_LABEL,  
    FR_RIBBON_EDIT,  
    FR_RIBBON_CHECKBOX,  
    FR_RIBBON_RADIOBUTTON,  
    FR_RIBBON_COMBOBOX,  
    FR_RIBBON_FONTCOMBOBOX,  
    FR_RIBBON_PALETTEBUTTON,  
    FR_RIBBON_LISTBUTTON,  
    FR_RIBBON_COLORBUTTON,
```

```
    FR\_RIBBON\_SLIDER,  
    FR\_RIBBON\_SEPARATOR /* The type is separator. */  
};
```

**Description**

The type of ribbon element.

**Head file reference**

fr\_barExpT.h: 464

**FR\_RIBBON\_NULL**

Unknown type.

**FR\_RIBBON\_BUTTON**

The type is button.

**FR\_RIBBON\_LABEL**

The type is label.

**FR\_RIBBON\_EDIT**

The type is edit.

**FR\_RIBBON\_CHECKBOX**

The type is check box.

**FR\_RIBBON\_RADIOBUTTON**

The type is radio button.

**FR\_RIBBON\_COMBOBOX**

The type is combo box.

**FR\_RIBBON\_FONTCOMBOBOX**

The type is font combo box.

**FR\_RIBBON\_PALETTEBUTTON**

The type is palette button.

**FR\_RIBBON\_LISTBUTTON**

The type is list button.

**FR\_RIBBON\_COLORBUTTON**

The type is color button.

**FR\_RIBBON\_SLIDER**

The type is slider.

**FR\_RIBBON\_SEPARATOR** /\* The type is separator. \*/

## Functions

## Functions summary

### [\*\*FRRibbonElementAddChangeImage\*\*](#)

Adds the 32\*32 bitmap and the 16\*16 bitmap to the bitmap list that can be used to change the icon of the ribbon element. This interface is only applicable for [FR\\_RIBBON\\_BUTTON](#) , [FR\\_RIBBON\\_EDIT](#) , [FR\\_RIBBON\\_LISTBUTTON](#) , [FR\\_RIBBON\\_COLORBUTTON](#) , and [FR\\_RIBBON\\_PALETTEBUTTON](#) .

### [\*\*FRRibbonElementAddSubItem\*\*](#)

Add a new sub item to the ribbon element.

### [\*\*FRRibbonElementChangeImage\*\*](#)

Changes the icon of the ribbon element.

### [\*\*FRRibbonElementCopyElementAsSubItem\*\*](#)

Copies an existing ribbon element as the sub item.

### [\*\*FRRibbonElementDoExecuteProc\*\*](#)

Executes the [FRExecuteProc](#) () associated with the ribbon element.

### [\*\*FRRibbonElementFindSubElementByIndex\*\*](#)

Gets the ribbon element by index.

### [\*\*FRRibbonElementGetAccel\*\*](#)

Gets the accelerator key of the ribbon element.

### [\*\*FRRibbonElementGetButtonMapId\*\*](#)

Gets the map ID.

### [\*\*FRRibbonElementGetCategoryName\*\*](#)

Gets the category name of the specified element.

### [\*\*FRRibbonElementGetClientData\*\*](#)

Gets the client data.

### [\*\*FRRibbonElementGetClientRect\*\*](#)

Gets the client rectangle of the ribbon element.

### [\*\*FRRibbonElementGetCorrespondingButton\*\*](#)

Gets the corresponding button according to the type.

### [\*\*FRRibbonElementGetDescription\*\*](#)

Gets the description of the ribbon element.

### [\*\*FRRibbonElementGetKey\*\*](#)

Gets the shortcut key of the ribbon element.

### [\*\*FRRibbonElementGetName\*\*](#)

Gets the name of the ribbon element.

### [\*\*FRRibbonElementGetOriginalID\*\*](#)

Gets the original ID of the ribbon element.

### [\*\*FRRibbonElementGetPanelName\*\*](#)

Gets the panel name of the specified element.

### [\*\*FRRibbonElementGetSubElementByName\*\*](#)

Gets the ribbon element by name.

### [\*\*FRRibbonElementGetSubItemCount\*\*](#)

Gets the count of the sub items.

### [\*\*FRRibbonElementGetTitle\*\*](#)

Gets the title of the ribbon element.

### [\*\*FRRibbonElementGetTooltip\*\*](#)

Gets the tooltip of the ribbon element.

### [\*\*FRRibbonElementGetType\*\*](#)

Gets the type of the ribbon element.

### [\*\*FRRibbonElementGetVisible\*\*](#)

Checks whether the ribbon element is visible or not.

**FRRibbonElementGetWindowRect**

Gets the window rectangle of the ribbon element.

**FRRibbonElementIsBackstageViewTabElementKeepState**

Checks whether to keep the state of element when the back stage view is open.

**FRRibbonElementIsEnabled**

Checks whether the ribbon element is enabled or not.

**FRRibbonElementIsMarked**

Checks whether the ribbon element is marked or not.

**FRRibbonElementRemove**

Removes the ribbon element.

**FRRibbonElementSetAccel**

Sets the accelerator key of the ribbon element.

**FRRibbonElementSetAsSubElement**

Sets that the ribbon element uses the specified sub item's response function.

**FRRibbonElementSetBackstageViewTabElementKeepState**

Sets to keep the state of element when the back stage view is open.

**FRRibbonElementSetClientData**

Sets the client data to the ribbon element.

**FRRibbonElementSetComputeEnabledProc**

Sets the callback function that will be called to determine whether the ribbon element is enabled or not.

**FRRibbonElementSetComputeMarkedProc**

Sets the callback function that will be called to determine whether the ribbon element is marked or not.

**FRRibbonElementSetDescription**

Sets the description of the ribbon element..

**FRRibbonElementSetDropdownProc**

Sets the drop-down callback function.

**FRRibbonElementSetExecuteProc**

Sets the callback function that will be called when the user clicks the ribbon element.

**FRRibbonElementSetImage**

Sets the icon of the ribbon element. The ribbon element contains a large icon and a small icon.

**FRRibbonElementSetImageInitProc**

Sets the callback invoked by Foxit Reader to init the visible image.

**FRRibbonElementSetImplicitLargeBitmap**

Sets the implicit large icon. It is the default large icon, when the icon needs to be changed by [FRRibbonElementSetAsSubElement](#) . If the ribbon element already has the large icon, this interface is ignored.

**FRRibbonElementSetKey**

Sets the shortcut key of the ribbon element.

**FRRibbonElementSetSelectOnly**

Sets whether the element is just only for selecting or not.

**FRRibbonElementSetShowMode**

Sets the showing mode.

**FRRibbonElementSetTitle**

Sets the title of the ribbon element.

**FRRibbonElementSetTooltip**

Sets the tooltip of the ribbon element.

**FRRibbonElementSetTooltipImage**

Sets the icon of the tooltip of the ribbon element.

**[FRRibbonElementSetTooltipImageII](#)**

Sets the icon of the tooltip of the ribbon element.

**[FRRibbonElementSetVisible](#)**

Sets the ribbon element to be visible or not.

**[FRRibbonElementShowInQATCustomizeToolsDlg](#)**

Sets whether the ribbon element can be shown in the QAT customize tools dialog.

**Functions detail****[FRRibbonElementAddChangeImage](#)****Syntax**

```
void FRRibbonElementAddChangeImage (
    FR\_RibbonElement ribbonElement,
    FS\_DIBitmap largeBitmap,
    FS\_DIBitmap smallBitmap,
    FS\_INT32* nLargeIndex,
    FS\_INT32* nSmallIndex
);
```

**Description**

Adds the 32\*32 bitmap and the 16\*16 bitmap to the bitmap list that can be used to change the icon of the ribbon element. This interface is only applicable for

[FR\\_RIBBON\\_BUTTON](#) , [FR\\_RIBBON\\_EDIT](#) , [FR\\_RIBBON\\_LISTBUTTON](#) ,  
[FR\\_RIBBON\\_COLORBUTTON](#) , and [FR\\_RIBBON\\_PALETTEBUTTON](#) .

**Parameter**

ribbonElement	[In] The input ribbon element object.
largeBitmap	[In] The input 32*32 bitmap list.
smallBitmap	[In] The input 16*16 bitmap list.
nLargeIndex	[Out] It receives the index of large bitmap added to the bitmap list.

  

nSmallIndex	[Out] It receives the index of small bitmap added to the bitmap list.
-------------	---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2166

**Related method**

[FRRibbonElementChangeImage](#)

**Since**  
[SDK LATEEST VERSION > 2.0](#)

**FRRibbonElementAddSubItem****Syntax**

```
void* FRRibbonElementAddSubItem (
    FR\_RibbonElement ribbonElement,
    FR\_Ribbon\_Element\_Type nElementType,
    const FS\_CHAR* name,
    FS\_LPCWSTR lpwsTitle,
    FS\_INT32 nPos,
    FS\_BOOL bOnTop,
    FS\_BOOL bChangeFun,
    FS\_BOOL bChangeImage
);
```

**Description**

Add a new sub item to the ribbon element.

**Parameter**


---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

nElementType	[In] The input type of ribbon element.
--------------	--

---

name	[In] The input name of ribbon element.
------	--

---

lpwsTitle	[In] The input title of ribbon element.
-----------	---

---

nPos	[In] The input position of ribbon element.
------	--

---

bOnTop	[In] Whether the added sub item is on the top or not.
--------	---

---

bChangeFun	[In] When the user clicks the sub item, whether the parent's function should be change to the sub item's. Sets it FALSE as default.
------------	---

---

bChangeImage	[In] When the user clicks the sub item, whether the parent's icon should be change to the sub item's. Sets it FALSE as default.
--------------	---

---

**Return**

The new sub item of the ribbon element. For example, if sets *nElementType* as [FR\\_RIBBON\\_BUTTON](#), the returned value can be converted to [FR.RibbonButton](#).

**Head file reference**

fr\_barTempl.h: 1915

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

[FRRibbonElementChangeImage](#)

**Syntax**

```
void FRRibbonElementChangeImage (
    FR.RibbonElement ribbonElement,
    FS\_BOOL bUseDefault,
    FS\_INT32 nLargeIndex,
    FS\_INT32 nSmallIndex
);
```

**Description**

Changes the icon of the ribbon element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

bUseDefault	[In] <a href="#">TRUE</a> if uses the original icon, otherwise uses the icon of specified index.
-------------	--

---

nLargeIndex	[In] The specified index of the large icon.
-------------	---

---

nSmallIndex	[In] The specified index of the small icon.
-------------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2178

**Related method**

[FRRibbonElementAddChangeImage](#)

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonElementCopyElementAsSubItem****Syntax**

```
void* FRRibbonElementCopyElementAsSubItem (
    FR.RibbonElement ribbonElement,
    FR.RibbonElement srcElement,
    FS\_BOOL bChangeFun,
    FS\_BOOL bChangeImage,
    FS\_BOOL bAlwaysSmall
);
```

**Description**

Copies an existing ribbon element as the sub item.

**Parameter**

ribbonElement	[In] The input ribbon element object.
srcElement	[In] The input ribbon element to be copied.
bChangeFun	[In] When the user clicks the sub item, whether the parent's function should be change to the sub item's. Sets it FALSE as default.
bChangeImage	[In] When the user clicks the sub item, whether the parent's icon should be change to the sub item's. Sets it FALSE as default.
bAlwaysSmall	[In] Sets it as <a href="#">TRUE</a> if the sub item is always shown as small.

**Return**

The sub item of the ribbon element. If the type is [FR.Ribbon.BUTTON](#) , the returned value can be converted to [FR.RibbonButton](#) .

**Head file reference**

fr\_barTempl.h: 2325

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonElementDoExecuteProc****Syntax**

```
void FRRibbonElementDoExecuteProc (
    FR.RibbonElement ribbonElement
);
```

**Description**

Executes the [FRExecuteProc](#) () associated with the ribbon element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1970

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonElementFindSubElementByIndex****Syntax**

```
FR_RibbonElement FRRibbonElementFindSubElementByIndex (
    FR\_RibbonElement ribbonElement,
    FS\_INT32 nIndex
);
```

**Description**

Gets the ribbon element by index.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

nIndex	[In] The specified index.
--------	---------------------------

---

**Return**

The found ribbon element.

**Head file reference**

fr\_barTempl.h: 2301

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonElementGetAccel****Syntax**

```
void FRRibbonElementGetAccel (
    FR\_RibbonElement ribbonElement,
    FS\_BOOL* bAlt,
    FS\_BOOL* bShift,
    FS\_BOOL* bCtrl,
    FS\_CHAR* key
);
```

**Description**

Gets the accelerator key of the ribbon element.

**Parameter**

ribbonElement	[In] The input ribbon element object.
bAlt	[Out] It receives <a href="#">TRUE</a> if the <i>Alt</i> key is pressed down.
bShift	[Out] It receives <a href="#">TRUE</a> if the <i>Shift</i> key is pressed down.
bCtrl	[Out] It receives <a href="#">TRUE</a> if the <i>Ctrl</i> key is pressed down.
key	[Out] It receives the accelerator key

**Return**

void

**Head file reference**

fr\_barTempl.h: 2042

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonElementGetButtonMapId****Syntax**

```
DWORD FRRibbonElementGetButtonMapId (
    FR\_RibbonElement ribbonElement
);
```



**Description**

Gets the map ID.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

**Return**

The map ID.

**Head file reference**

fr\_barTempl.h: 2449

**Related method****Since**

[SDK LATEEST VERSION > 3.0.0.0](#)

**FRRibbonElementGetCategoryName****Syntax**

```
void FRRibbonElementGetCategoryName (
    FR\_RibbonElement ribbonElement,
    FS\_ByteString* outName
);
```

**Description**

Gets the category name of the specified element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

outName	[Out] (Filled by the method) A string buffer to receive the name.
---------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 2495

**Related method****Since**

[SDK LATEEST VERSION > 7.3](#)

## FRRibbonElementGetClientData

### Syntax

```
void* FRRibbonElementGetClientData (
    FR.RibbonElement ribbonElement
);
```

### Description

Gets the client data.

### Parameter

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

### Return

The client data.

### Head file reference

fr\_barTempl.h: 2016

### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonElementGetClientRect

### Syntax

```
FS_BOOL FRRibbonElementGetClientRect (
    FR.RibbonElement ribbonElement,
    FS.Rect* rcClient
);
```

### Description

Gets the client rectangle of the ribbon element.

### Parameter

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

rcClient	[Out] It receives the client rectangle.
----------	---

---

### Return

TRUE for success, otherwise failure.

### Head file reference

fr\_barTempl.h: 2401



**Related method**[FRRibbonElementGetWindowRect](#)**Since**[SDK LATEEST VERSION > 2.1.0.3](#)[FRRibbonElementGetCorrespondingButton](#)**Syntax**

```
void* FRRibbonElementGetCorrespondingButton (
    FR\_RibbonElement ribbonElement
);
```

**Description**

Gets the corresponding button according to the type.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

**Return**

The ribbon button. If the type is [FR\\_RIBBON\\_BUTTON](#), the returned value can be converted to [FR.RibbonButton](#).

**Head file reference**

fr\_barTempl.h: 2340

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)[FRRibbonElementGetDescription](#)**Syntax**

```
void FRRibbonElementGetDescription (
    FR\_RibbonElement ribbonElement,
    FS\_WideString* wsDes
);
```

**Description**

Gets the description of the ribbon element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---



wsDes	[Out] It receives the description of the ribbon element.
-------	--

**Return**

void

**Head file reference**

fr\_barTempl.h: 2117

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonElementGetKey****Syntax**

```
void FRRibbonElementGetKey (
    FR\_RibbonElement ribbonElement,
    FS\_WideString* wsKey
);
```

**Description**

Gets the shortcut key of the ribbon element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

wsKey	[Out] It receives the shortcut key of the ribbon element.
-------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2069

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonElementGetName****Syntax**

```
void FRRibbonElementGetName (
    FR\_RibbonElement ribbonElement,
    FS\_ByteString* outName
);
```

**Description**

Gets the name of the ribbon element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

outName	[Out] It receives the name of the ribbon element.
---------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2278

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonElementGetOriginalID****Syntax**

```
FS_DWORD FRRibbonElementGetOriginalID (
    FR.RibbonElement ribbonElement
);
```

**Description**

Gets the original ID of the ribbon element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

**Return**

The original ID of the ribbon element.

**Head file reference**

fr\_barTempl.h: 2267

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)



**FRRibbonElementGetPanelName****Syntax**

```
void FRRibbonElementGetPanelName (
    FR\_RibbonElement ribbonElement,
    FS\_ByteString* outName
);
```

**Description**

Gets the panel name of the specified element.

**Parameter**

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

outName	[Out] (Filled by the method) A string buffer to receive the name.
---------	---

**Return**

void.

**Head file reference**

[fr\\_barTempl.h](#): 2507

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.3](#)

**FRRibbonElementGetSubElementByName****Syntax**

```
FR\_RibbonElement FRRibbonElementGetSubElementByName (
    FR\_RibbonElement ribbonElement,
    const FS\_CHAR* name
);
```

**Description**

Gets the ribbon element by name.

**Parameter**

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

name	[In] The specified name.
------	--------------------------

**Return**

The found ribbon element.

**Head file reference**

fr\_barTempl.h: 2313

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonElementGetSubItemCount****Syntax**

```
FS_INT32 FRRibbonElementGetSubItemCount (
    FR.RibbonElement ribbonElement
);
```

**Description**

Gets the count of the sub items.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

**Return**

The count of the sub items.

**Head file reference**

fr\_barTempl.h: 2290

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonElementGetTitle****Syntax**

```
void FRRibbonElementGetTitle (
    FR.RibbonElement ribbonElement,
    FS.WideString* wsTitle
);
```

**Description**

Gets the title of the ribbon element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

wsTitle	[Out] It receives the title of the ribbon element.
---------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2141

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonElementGetTooltip****Syntax**

```
void FRRibbonElementGetTooltip (
    FR\_RibbonElement ribbonElement,
    FS\_WideString* wsTooltip
);
```

**Description**

Gets the tooltip of the ribbon element.

**Parameter**


---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

wsTooltip	[Out] It receives the tooltip of the ribbon element.
-----------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2093

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonElementGetType****Syntax**

```
FR_Ribbon_Element_Type FRRibbonElementGetType (
    FR\_RibbonElement ribbonElement
);
```

**Description**

Gets the type of the ribbon element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

**Return**

The type of the ribbon element.

**Head file reference**

fr\_barTempl.h: 2256

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonElementGetVisible****Syntax**

```
FS_BOOL FRRibbonElementGetVisible (
    FR\_RibbonElement ribbonElement
);
```

**Description**

Checks whether the ribbon element is visible or not.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

**Return**

[TRUE](#) if the ribbon element is visible, otherwise [FALSE](#).

**Head file reference**

fr\_barTempl.h: 2234

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonElementGetWindowRect****Syntax**

```
FS_BOOL FRRibbonElementGetWindowRect (
    FR.RibbonElement ribbonElement,
    FS.Rect* rcScreen
);
```

**Description**

Gets the window rectangle of the ribbon element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

rcScreen	[Out] It receives the window rectangle.
----------	---

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fr\_barTempl.h: 2408

**Related method**

[FRRibbonElementGetClientRect](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)

**FRRibbonElementIsBackstageViewTabElementKeepState****Syntax**

```
FS_BOOL FRRibbonElementIsBackstageViewTabElementKeepState (
    FR.RibbonElement ribbonElement
);
```

**Description**

Checks whether to keep the state of element when the back stage view is open.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

**Return**

Whether to keep the state of element when the back stage view is open.

**Head file reference**

fr\_barTempl.h: 2472

#### Related method

##### Since

[SDK LATEEST VERSION > 3.0.0.0](#)

## FRRibbonElementIsEnabled

#### Syntax

```
FS_BOOL FRRibbonElementIsEnabled (
    FR\_RibbonElement ribbonElement
);
```

#### Description

Checks whether the ribbon element is enabled or not.

#### Parameter

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

#### Return

[TRUE](#) if the ribbon element is enabled, otherwise [FALSE](#).

#### Head file reference

fr\_barTempl.h: 1981

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

## FRRibbonElementIsMarked

#### Syntax

```
FS_BOOL FRRibbonElementIsMarked (
    FR\_RibbonElement ribbonElement
);
```

#### Description

Checks whether the ribbon element is marked or not.

#### Parameter

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

#### Return

[TRUE](#) if the ribbon element is marked, otherwise [FALSE](#).

**Head file reference**

fr\_barTempl.h: 1992

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonElementRemove****Syntax**

```
FS_BOOL FRRibbonElementRemove (
    FR.RibbonElement ribbonElement
);
```

**Description**

Removes the ribbon element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

**Return**

[TRUE](#) for success, otherwise failure.

**Head file reference**

fr\_barTempl.h: 2245

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonElementSetAccel****Syntax**

```
void FRRibbonElementSetAccel (
    FR.RibbonElement ribbonElement,
    FS_BOOL bAlt,
    FS_BOOL bShift,
    FS_BOOL bCtrl,
    FS_CHAR key
);
```

**Description**

Sets the accelerator key of the ribbon element.

**Parameter**


---

ribbonElement	[In] The input ribbon element object.
bAlt	[In] Whether the <i>Alt</i> key is pressed down or not.
bShift	[In] Whether the <i>Shift</i> key is pressed down or not.
bCtrl	[In] Whether the <i>Ctrl</i> key is pressed down or not.
key	[In] The input accelerator key.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2027

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonElementSetAsSubElement****Syntax**

```
void FRRibbonElementSetAsSubElement (
    FR\_RibbonElement ribbonElement,
    const FS\_CHAR* subElementName,
    FS\_BOOL bChangeImage
);
```

**Description**

Sets that the ribbon element uses the specified sub item's response function.

**Parameter**


---

ribbonElement	[In] The input ribbon element object.
subElementName	[In] The specified name.
bChangeImage	[In] Whether the parent's icon should be change to the sub item's.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2351

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonElementSetBackstageViewTabElementKeepState****Syntax**

```
void FRRibbonElementSetBackstageViewTabElementKeepState (
    FR\_RibbonElement ribbonElement,
    FS\_BOOL bKeepState
);
```

**Description**

Sets to keep the state of element when the back stage view is open.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

bKeepState	[In] Whether keeps the state of element when the back stage view is open.
------------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2460

**Related method****Since**

[SDK LATEEST VERSION > 3.0.0.0](#)

**FRRibbonElementSetClientData****Syntax**

```
void FRRibbonElementSetClientData (
    FR\_RibbonElement ribbonElement,
    void* clientData,
    FRFreeDataProc callback
```

```
 );
```

**Description**

Sets the client data to the ribbon element.

**Parameter**


---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

clientData	[In] The input client data.
------------	-----------------------------

---

callback	[In] The callback function will be called when the ribbon element is to be released.
----------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2003

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonElementSetComputeEnabledProc****Syntax**

```
void FRRibbonElementSetComputeEnabledProc (
    FR.RibbonElement ribbonElement,
    FRComputeEnabledProc proc
);
```

**Description**

Sets the callback function that will be called to determine whether the ribbon element is enabled or not.

**Parameter**


---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

proc	[In] The callback will be called to determine whether the ribbon element is enabled or not.
------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1946

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonElementSetComputeMarkedProc****Syntax**

```
void FRRibbonElementSetComputeMarkedProc (
    FR\_RibbonElement ribbonElement,
    FRCComputeMarkedProc proc
);
```

**Description**

Sets the callback function that will be called to determine whether the ribbon element is marked or not.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

proc	[In] The callback will be called to determine whether the ribbon element is marked or not.
------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1958

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonElementSetDescription****Syntax**

```
void FRRibbonElementSetDescription (
    FR\_RibbonElement ribbonElement,
    FS\_LPCWSTR lpwsDes
);
```

**Description**

Sets the description of the ribbon element..

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
lpwsDes	[In] The input description of the ribbon element.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2105

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonElementSetDropdownProc****Syntax**

```
void FRRibbonElementSetDropdownProc (
    FR\_RibbonElement ribbonElement,
    FRDropDownProc proc
);
```

**Description**

Sets the drop-down callback function.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
proc	[In] The callback function is called when the user clicks the ribbon element drop-down arrow.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2389

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonElementSetExecuteProc****Syntax**

```
void FRRibbonElementSetExecuteProc (
    FR\_RibbonElement ribbonElement,
    FRExecuteProc proc
);
```

**Description**

Sets the callback function that will be called when the user clicks the ribbon element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
proc	[In] The callback will be called when the user clicks the ribbon element.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1934

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonElementSetImage****Syntax**

```
void FRRibbonElementSetImage (
    FR\_RibbonElement ribbonElement,
    FS\_DIBitmap largeBitmap,
    FS\_DIBitmap smallBitmap
);
```

**Description**

Sets the icon of the ribbon element. The ribbon element contains a large icon and a small icon.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

---

largeBitmap	[In] The input 32*32 bitmap.
-------------	------------------------------

---

smallBitmap	[In] The input 16*16 bitmap.
-------------	------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2153

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonElementSetImageInitProc****Syntax**

```
void FRRibbonElementSetImageInitProc (
    FR\_RibbonElement ribbonElement,
    FRRibbonElementImageInitProc proc
);
```

**Description**

Sets the callback invoked by Foxit Reader to init the visible image.

**Parameter**


---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

proc	[In] The callback invoked by Foxit Reader to init the visible image.
------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2437

**Related method****Since**[SDK LATEEST VERSION > 2.1.0.4](#)**FRRibbonElementSetImplicitLargeBitmap**

**Syntax**

```
void FRRibbonElementSetImplicitLargeBitmap (
    FR\_RibbonElement ribbonElement,
    FS\_DIBitmap largeBitmap
);
```

**Description**

Sets the implicit large icon. It is the default large icon, when the icon needs to be changed by [FRRibbonElementSetAsSubElement](#). If the ribbon element already has the large icon, this interface is ignored.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

largeBitmap	[In] The input implicit large icon.
-------------	-------------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2364

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonElementSetKey****Syntax**

```
void FRRibbonElementSetKey (
    FR\_RibbonElement ribbonElement,
    FS\_LPCWSTR lpwsShorcutKey
);
```

**Description**

Sets the shortcut key of the ribbon element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

lpwsShorcutKey	[In] The input shortcut key.
----------------	------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2057

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonElementSetSelectOnly****Syntax**

```
void FRRibbonElementSetSelectOnly (
    FR\_RibbonElement ribbonElement,
    FS\_BOOL bSelectOnly
);
```

**Description**

Sets whether the element is just only for selecting or not.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

bSelectOnly	[In] Whether the element is just only for selecting or not.
-------------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 2483

**Related method****Since**[SDK LATEEST VERSION > 7.2.2](#)**FRRibbonElementSetShowMode****Syntax**

```
void FRRibbonElementSetShowMode (
    FR\_RibbonElement ribbonElement,
    FS\_BOOL bLarge
);
```

**Description**

Sets the showing mode.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
bLarge	[In] <a href="#">TRUE</a> if the ribbon element is shown using the large icon.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2377

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonElementSetTitle****Syntax**

```
void FRRibbonElementSetTitle (
    FR\_RibbonElement ribbonElement,
    FS\_LPCWSTR lpwsTitle
);
```

**Description**

Sets the title of the ribbon element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
lpwsTitle	[In] The input title of the ribbon element.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2129

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonElementSetTooltip

### Syntax

```
void FRRibbonElementSetTooltip (
    FR\_RibbonElement ribbonElement,
    FS\_LPCWSTR lpwsTooltip
);
```

### Description

Sets the tooltip of the ribbon element.

### Parameter

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

lpwsTooltip	[In] The input tooltip of the ribbon element.
-------------	---

---

### Return

void

### Head file reference

fr\_barTempl.h: 2081

### Related method

#### Since

[SDK LATEEST VERSION > 2.0](#)

## FRRibbonElementSetTooltipImage

### Syntax

```
void FRRibbonElementSetTooltipImage (
    FR\_RibbonElement ribbonElement,
    FS\_DIBitmap bitmap
);
```

### Description

Sets the icon of the tooltip of the ribbon element.

### Parameter

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

bitmap	[In] The input icon of the tooltip.
--------	-------------------------------------

---

### Return

void



**Head file reference**

fr\_barTempl.h: 2197

**Related method**[FRRibbonElementSetTooltipImageII](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonElementSetTooltipImageII****Syntax**

```
void FRRibbonElementSetTooltipImageII (
    FR\_RibbonElement ribbonElement,
    HINSTANCE hInstance,
    unsigned int nImageID
);
```

**Description**

Sets the icon of the tooltip of the ribbon element.

**Parameter**

---

ribbonElement	[In] The input ribbon element object.
---------------	---------------------------------------

---

hInstance	[In] The instance handle of the plug-in.
-----------	--

---

nImageID	[In] The resource ID of the bitmap.
----------	-------------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2204

**Related method**[FRRibbonElementSetTooltipImage](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonElementSetVisible****Syntax**

```
void FRRibbonElementSetVisible (
    FR\_RibbonElement ribbonElement,
```

---

```
BOOL bVisible  
);
```

**Description**

Sets the ribbon element to be visible or not.

**Parameter**


---

ribbonElement	[In] The input ribbon element object.
bVisible	[In] <a href="#">TRUE</a> if the ribbon element is visible, otherwise <a href="#">FALSE</a> .

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2222

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonElementShowInQATCustomizeToolsDlg****Syntax**

```
void FRRibbonElementShowInQATCustomizeToolsDlg (  
    FR_RibbonElement ribbonElement,  
    FS_BOOL bShow  
) ;
```

**Description**

Sets whether the ribbon element can be shown in the QAT customize tools dialog.

**Parameter**


---

ribbonElement	[In] The input ribbon element object.
bShow	[In] It indicates whether the ribbon element can be shown in the QAT customize tools dialog.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 2425

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)

## FR\_RibbonFontComboBox

**Used by**

### Description

The ribbon font combo box is a type of ribbon element.

**Used by**

[\*\*FRRibbonFontComboBoxAddFont\*\*](#)  
[\*\*FRRibbonFontComboBoxEnableDropDownListResize\*\*](#)  
[\*\*FRRibbonFontComboBoxGetCurSel\*\*](#)  
[\*\*FRRibbonFontComboBoxGetEditText\*\*](#)  
[\*\*FRRibbonFontComboBoxGetElement\*\*](#)  
[\*\*FRRibbonFontComboBoxGetFontCount\*\*](#)  
[\*\*FRRibbonFontComboBoxGetFontIndex\*\*](#)  
[\*\*FRRibbonFontComboBoxGetFontName\*\*](#)  
[\*\*FRRibbonFontComboBoxGetHWnd\*\*](#)  
[\*\*FRRibbonFontComboBoxGetItem\*\*](#)  
[\*\*FRRibbonFontComboBoxGetScriptName\*\*](#)  
[\*\*FRRibbonFontComboBoxInsertFont\*\*](#)  
[\*\*FRRibbonFontComboBoxRemoveFont\*\*](#)  
[\*\*FRRibbonFontComboBoxRemoveFont2\*\*](#)  
[\*\*FRRibbonFontComboBoxSelectItem\*\*](#)  
[\*\*FRRibbonFontComboBoxSelectItem2\*\*](#)  
[\*\*FRRibbonFontComboBoxSetEditBox\*\*](#)  
[\*\*FRRibbonFontComboBoxSetEditText\*\*](#)  
[\*\*FRRibbonFontComboBoxSetFocus\*\*](#)  
[\*\*FRRibbonFontComboBoxSetWidth\*\*](#)

### Functions

#### Functions summary

[\*\*FRRibbonFontComboBoxAddFont\*\*](#)

Adds a font into the ribbon font combo box.

[\*\*FRRibbonFontComboBoxEnableDropDownListResize\*\*](#)

Sets whether the dropped down list can be resized or not.

[\*\*FRRibbonFontComboBoxGetCurSel\*\*](#)

Gets the index of the currently selected item.

[\*\*FRRibbonFontComboBoxGetEditText\*\*](#)

Gets the edit text of the ribbon font combo box.

[\*\*FRRibbonFontComboBoxGetElement\*\*](#)

Gets the ribbon element associated with the ribbon font combo box. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

#### [\*\*FRRibbonFontComboBoxGetFontCount\*\*](#)

Gets the font count of the ribbon font combo box.

#### [\*\*FRRibbonFontComboBoxGetFontIndex\*\*](#)

Gets the index of the font.

#### [\*\*FRRibbonFontComboBoxGetFontName\*\*](#)

Gets the font name.

#### [\*\*FRRibbonFontComboBoxGetHWnd\*\*](#)

Gets the window handle of the ribbon font combo box.

#### [\*\*FRRibbonFontComboBoxGetItem\*\*](#)

Gets the text of the item by index.

#### [\*\*FRRibbonFontComboBoxGetScriptName\*\*](#)

Gets the font script name.

#### [\*\*FRRibbonFontComboBoxInsertFont\*\*](#)

Inserts a font into the ribbon font combo box.

#### [\*\*FRRibbonFontComboBoxRemoveFont\*\*](#)

Remove the font by font name.

#### [\*\*FRRibbonFontComboBoxRemoveFont2\*\*](#)

Remove the font by font index.

#### [\*\*FRRibbonFontComboBoxSelectItem\*\*](#)

Selects the specified item by index.

#### [\*\*FRRibbonFontComboBoxSelectItem2\*\*](#)

Selects the item by font name.

#### [\*\*FRRibbonFontComboBoxSetEditBox\*\*](#)

Sets whether the font combo box to have edit box or not.

#### [\*\*FRRibbonFontComboBoxSetEditText\*\*](#)

Sets the edit text of the ribbon font combo box.

#### [\*\*FRRibbonFontComboBoxSetFocus\*\*](#)

Sets the focus to the ribbon font combo box or not.

#### [\*\*FRRibbonFontComboBoxSetWidth\*\*](#)

Sets the width of the ribbon font combo box.

## Functions detail

### [\*\*FRRibbonFontComboBoxAddFont\*\*](#)

#### Syntax

```
FS_INT32 FRRibbonFontComboBoxAddFont (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_LPCWSTR lpwsFaceName,
    FS\_BOOL bSort,
    FS\_BYTE nCharSet
);
```

#### Description

Adds a font into the ribbon font combo box.

#### Parameter

---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

lpwsFaceName	[In] The input font face name.
--------------	--------------------------------

---

bSort	[In] Whether to sort the font name or not.
-------	--

---

nCharSet	[In] The input charset. Sets it DEFAULT_CHARSET as default.
----------	---

---

**Return**

The index of the font added into the ribbon font combo box.

**Head file reference**

fr\_barTempl.h: 3162

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRRibbonFontComboBoxEnableDropDownListResize****Syntax**

```
void FRRibbonFontComboBoxEnableDropDownListResize (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_BOOL bEnable
);
```

**Description**

Sets whether the dropped down list can be resized or not.

**Parameter**


---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

bEnable	[In] <a href="#">TRUE</a> if the dropped down list can be resized.
---------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3139

**Related method****Since**

## [SDK\\_LATEEST\\_VERSION > 2.0](#)

### FRRibbonFontComboBoxGetCurSel

#### Syntax

```
FS_INT32 FRRibbonFontComboBoxGetCurSel (
    FR\_RibbonFontComboBox ribbonFontComboBox
);
```

#### Description

Gets the index of the currently selected item.

#### Parameter

---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

#### Return

The index of the currently selected item.

#### Head file reference

fr\_barTempl.h: 3068

#### Related method

#### Since

## [SDK\\_LATEEST\\_VERSION > 2.0](#)

### FRRibbonFontComboBoxGetEditText

#### Syntax

```
void FRRibbonFontComboBoxGetEditText (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_WideString* wsText
);
```

#### Description

Gets the edit text of the ribbon font combo box.

#### Parameter

---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

wsText	[Out] It receives the edit text.
--------	----------------------------------

---

#### Return

void

**Head file reference**

fr\_barTempl.h: 3115

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonFontComboBoxGetElement****Syntax**

```
FR_RibbonElement FRRibbonFontComboBoxGetElement (
    FR\_RibbonFontComboBox ribbonFontComboBox
);
```

**Description**

Gets the ribbon element associated with the ribbon font combo box. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**Parameter**

---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

**Return**

The ribbon element associated with the ribbon font combo box.

**Head file reference**

fr\_barTempl.h: 3043

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonFontComboBoxGetFontCount****Syntax**

```
FS_INT32 FRRibbonFontComboBoxGetFontCount (
    FR\_RibbonFontComboBox ribbonFontComboBox
);
```

**Description**

Gets the font count of the ribbon font combo box.

**Parameter**

---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

**Return**

The font count of the ribbon font combo box.

**Head file reference**

fr\_barTempl.h: 3264

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

FRRibbonFontComboBoxGetFontIndex

**Syntax**

```
FS_INT32 FRRibbonFontComboBoxGetFontIndex (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_LPCWSTR lpwsFaceName
);
```

**Description**

Gets the index of the font.

**Parameter**


---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

lpwsFaceName	[In] The input font face name.
--------------	--------------------------------

---

**Return**

The index of the font.

**Head file reference**

fr\_barTempl.h: 3190

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

FRRibbonFontComboBoxGetFontName

**Syntax**

```
void FRRibbonFontComboBoxGetFontName (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_INT32 nIndex,
    FS\_WideString* outName
```

);

**Description**

Gets the font name.

**Parameter**

---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

nIndex	[In] The input font index.
--------	----------------------------

---

outName	[Out] It receives the font name.
---------	----------------------------------

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 3202

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRRibbonFontComboBoxGetHWnd****Syntax**

```
HWND FRRibbonFontComboBoxGetHWnd (
    FR\_RibbonFontComboBox ribbonFontComboBox
);
```

**Description**

Gets the window handle of the ribbon font combo box.

**Parameter**

---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

**Return**

The window handle of the ribbon font combo box.

**Head file reference**

fr\_barTempl.h: 3151

**Related method**

**Since**[SDK\\_LATEEST\\_VERSION > 3.0](#)**FRRibbonFontComboBoxGetItem****Syntax**

```
void FRRibbonFontComboBoxGetItem (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_INT32 nIndex,
    FS\_WideString* wsText
);
```

**Description**

Gets the text of the item by index.

**Parameter**

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

nIndex	[In] The specified index.
--------	---------------------------

wsText	[Out] It receives the text of the item.
--------	---

**Return**

void

**Head file reference**

[fr\\_barTempl.h: 3055](#)

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonFontComboBoxGetScriptName****Syntax**

```
void FRRibbonFontComboBoxGetScriptName (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_INT32 nIndex,
    FS\_WideString* outName
);
```

**Description**

Gets the font script name.

**Parameter**

---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

nIndex	[In] The input font index.
--------	----------------------------

---

outName	[Out] It receives the font script name.
---------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 3215

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.2.2](#)**FRRibbonFontComboBoxInsertFont****Syntax**

```
FS_INT32 FRRibbonFontComboBoxInsertFont (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_LPCWSTR lpwsFaceName,
    FS\_INT32 nIndex,
    FS\_BYTE nCharSet
);
```

**Description**

Inserts a font into the ribbon font combo box.

**Parameter**


---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

lpwsFaceName	[In] The input font face name.
--------------	--------------------------------

---

nIndex	[In] The position where the font will be inserted.
--------	--

---

nCharSet	[In] The input charset. Sets it DEFAULT_CHARSET as default.
----------	---

---

**Return**

The index of the font inserted into the ribbon font combo box.

**Head file reference**

---

fr\_barTempl.h: 3176

#### Related method

##### Since

[SDK LATEEST VERSION > 7.2.2](#)

## FRRibbonFontComboBoxRemoveFont

#### Syntax

```
FS_BOOL FRRibbonFontComboBoxRemoveFont (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_LPCWSTR lpwsFaceName
);
```

#### Description

Remove the font by font name.

#### Parameter

---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

lpwsFaceName	[In] The input font face name.
--------------	--------------------------------

---

#### Return

TRUE for success, otherwise not.

#### Head file reference

fr\_barTempl.h: 3240

#### Related method

##### Since

[SDK LATEEST VERSION > 7.2.2](#)

## FRRibbonFontComboBoxRemoveFont2

#### Syntax

```
FS_BOOL FRRibbonFontComboBoxRemoveFont2 (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_INT32 nIndex
);
```

#### Description

Remove the font by font index.

#### Parameter

---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

nIndex	[In] The input font index.
--------	----------------------------

---

**Return**

TRUE for success, otherwise not.

**Head file reference**

fr\_barTempl.h: 3252

**Related method****Since**

[SDK LATEEST VERSION > 7.2.2](#)

**FRRibbonFontComboBoxSelectItem****Syntax**

```
void FRRibbonFontComboBoxSelectItem (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_INT32 nItemIndex
);
```

**Description**

Selects the specified item by index.

**Parameter**


---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

nItemIndex	[In] The specified index.
------------	---------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3079

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonFontComboBoxSelectItem2****Syntax**

---

```
FS_BOOL FRRibbonFontComboBoxSelectItem2 (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_LPCWSTR lpwsFaceName
);
```

**Description**

Selects the item by font name.

**Parameter**


---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

lpwsFaceName	[In] The input font face name.
--------------	--------------------------------

**Return**

TRUE for success, otherwise not.

**Head file reference**

[fr\\_barTempl.h](#): 3228

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRRibbonFontComboBoxSetEditBox****Syntax**

```
void FRRibbonFontComboBoxSetEditBox (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_BOOL bHasEditBox
);
```

**Description**

Sets whether the font combo box to have edit box or not.

**Parameter**


---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

bHasEditBox	[In] TRUE if you want to set the font combo box to have edit box.
-------------	---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3103

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonFontComboBoxSetEditText****Syntax**

```
void FRRibbonFontComboBoxSetEditText (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_LPCWSTR lpwsText
);
```

**Description**

Sets the edit text of the ribbon font combo box.

**Parameter**

---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

lpwsText	[In] The input edit text.
----------	---------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3127

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonFontComboBoxSetFocus****Syntax**

```
void FRRibbonFontComboBoxSetFocus (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_BOOL bFocus
);
```

**Description**

Sets the focus to the ribbon font combo box or not.

**Parameter**

---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

bFocus	[In] Inputs TRUE to set the focus to the ribbon font combo box.
--------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 3275

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.3.1.0](#)**FRRibbonFontComboBoxSetWidth****Syntax**

```
void FRRibbonFontComboBoxSetWidth (
    FR\_RibbonFontComboBox ribbonFontComboBox,
    FS\_INT32 nWidth
);
```

**Description**

Sets the width of the ribbon font combo box.

**Parameter**

---

ribbonFontComboBox	[In] The input ribbon font combo box.
--------------------	---------------------------------------

---

nWidth	[In] The input width of the ribbon font combo box.
--------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3091

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FR\_RibbonLabel

### Used by

#### Description

The ribbon label is a type of ribbon element.

#### Used by

##### [FRRibbonLabelGetElement](#)

#### Functions

##### Functions summary

###### [FRRibbonLabelGetElement](#)

Gets the ribbon element associated with the ribbon label. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

##### Functions detail

###### [FRRibbonLabelGetElement](#)

###### Syntax

```
FR_RibbonElement FRRibbonLabelGetElement (
    FR_RibbonLabel ribbonLabel
);
```

###### Description

Gets the ribbon element associated with the ribbon label. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

###### Parameter

---

ribbonLabel	[In] The input ribbon label object.
-------------	-------------------------------------

---

###### Return

The ribbon element associated with the ribbon label.

###### Head file reference

fr\_barTempl.h: 2715

###### Related method

###### Since

[SDK LATEEST VERSION > 2.0](#)

## FR\_RibbonListButton

### Used by

#### Description

The ribbon list button is a type of ribbon element.

#### Used by

[FRRibbonListButtonAddGroup](#)  
[FRRibbonListButtonEnableResize](#)  
[FRRibbonListButtonGetElement](#)  
[FRRibbonListButtonGetGroupCount](#)  
[FRRibbonListButtonGetGroupTitle](#)  
[FRRibbonListButtonGetItemCount](#)  
[FRRibbonListButtonGetItemTitle](#)  
[FRRibbonListButtonGetItemToolTip](#)  
[FRRibbonListButtonGetSelectedItem](#)  
[FRRibbonListButtonIsGroupNameExist](#)  
[FRRibbonListButtonSetAlignedSide](#)  
[FRRibbonListButtonSetGroupTitle](#)  
[FRRibbonListButtonSetIconsInRow](#)  
[FRRibbonListButtonSetItemTitle](#)  
[FRRibbonListButtonSetItemToolTip](#)  
[FRRibbonListButtonSetSelectedItem](#)

#### Enumerations

##### Enumerations summary

###### **FR\_RIBBON\_LISTBUTTON\_ALIGNEDSIDE**

The alignment type of ribbon list button.

##### Enumerations detail

###### **FR\_RIBBON\_LISTBUTTON\_ALIGNEDSIDE**

###### Syntax

```
enum FR_RIBBON_LISTBUTTON_ALIGNEDSIDE{
    FR\_BUTTON\_ALIGN\_RIGHT,
    FR\_BUTTON\_ALIGN\_UP,
    FR\_BUTTON\_ALIGN\_DOWN /* Down alignment. */
};
```

###### Description

The alignment type of ribbon list button.

###### Head file reference

fr\_barExpT.h: 598

###### **FR\_BUTTON\_ALIGN\_RIGHT**

Right alignment.

**FR\_BUTTON\_ALIGN\_UP**

Up alignment.

**FR\_BUTTON\_ALIGN\_DOWN** /\* Down alignment. \*/

## Functions

### Functions summary

**FRRibbonListButtonAddGroup**

Add a new group to the ribbon list button.

**FRRibbonListButtonEnableResize**

Sets whether the ribbon list button can be resized or not.

**FRRibbonListButtonGetElement**

Gets the ribbon element associated with the ribbon list button. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**FRRibbonListButtonGetGroupCount**

Gets the group count of the ribbon list button.

**FRRibbonListButtonGetGroupName**

Gets the title of the group.

**FRRibbonListButtonGetItemCount**

Get the item count.

**FRRibbonListButtonGetItemTitle**

Gets the title of the item.

**FRRibbonListButtonGetItemToolTip**

Gets the tooltip of the item.

**FRRibbonListButtonGetSelectedItem**

Gets the info of the selected item.

**FRRibbonListButtonIsGroupNameExist**

Checks whether the group name exists or not.

**FRRibbonListButtonSetAlignedSide**

Sets the alignment type.

**FRRibbonListButtonSetGroupName**

Sets the title of the group.

**FRRibbonListButtonSetIconsInRow**

Sets the count of icons in a row.

**FRRibbonListButtonSetTitle**

Sets the title of the item.

**FRRibbonListButtonSetToolTip**

Sets the tooltip of the item.

**FRRibbonListButtonSetSelectedItem**

Sets the selected item.

### Functions detail

**FRRibbonListButtonAddGroup**

#### Syntax

---

```
void FRRibbonListButtonAddGroup (
    FR\_RibbonListButton ribbonListButton,
    const FS\_CHAR* name,
    FS\_LPCWSTR lpwsGroupTitle,
    FS\_DIBitmap paletteImages,
    FS\_INT32 nWidthForEachImage,
    FS\_WideStringArray arrLabels
);
```

**Description**

Add a new group to the ribbon list button.

**Parameter**

ribbonListButton	[In] The input ribbon list button object.
name	[In] The input group name.
lpwsGroupTitle	[In] The input group title.
paletteImages	[In] The input bitmap list of the group.
nWidthForEachImage	[In] The width of each bitmap.
arrLabels	[In] It contains labels of all the items.

**Return**

void

**Head file reference**

[fr\\_barTempl.h: 4141](#)

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

[FRRibbonListButtonEnableResize](#)

**Syntax**

```
void FRRibbonListButtonEnableResize (
    FR\_RibbonListButton ribbonListButton,
    FS\_BOOL bEnable,
    FS\_BOOL bVerticalOnly
);
```

**Description**

Sets whether the ribbon list button can be resized or not.

**Parameter**

ribbonListButton	[In] The input ribbon list button object.
bEnable	[In] TRUE if the ribbon list button can be resized.
bVerticalOnly	[In] TRUE if the ribbon list button can be resized in vertical direction only.

**Return**

void

**Head file reference**

fr\_barTempl.h: 4180

**Related method****Since**

[SDK LATEST VERSION > 2.0](#)

**FRRibbonListButtonGetElement****Syntax**

```
FR_RibbonElement FRRibbonListButtonGetElement (
    FR_RibbonListButton ribbonListButton
);
```

**Description**

Gets the ribbon element associated with the ribbon list button. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**Parameter**

ribbonListButton	[In] The input ribbon list button object.
------------------	---

**Return**

The ribbon element associated with the ribbon list button.

**Head file reference**

fr\_barTempl.h: 4129

**Related method**

**Since**  
[SDK LATEEST VERSION > 2.0](#)

### FRRibbonListButtonGetGroupCount

#### Syntax

```
FS_INT32 FRRibbonListButtonGetGroupCount (
    FR\_RibbonListButton ribbonListButton
);
```

#### Description

Gets the group count of the ribbon list button.

#### Parameter

---

ribbonListButton	[In] The input ribbon list button object.
------------------	---

---

#### Return

The group count of the ribbon list button.

#### Head file reference

fr\_barTempl.h: 4157

#### Related method

**Since**

[SDK LATEEST VERSION > 2.0](#)

### FRRibbonListButtonGetGroupTitle

#### Syntax

```
void FRRibbonListButtonGetGroupTitle (
    FR\_RibbonListButton ribbonListButton,
    const FS_CHAR* groupName,
    FS\_WideString* wsTitle
);
```

#### Description

Gets the title of the group.

#### Parameter

---

ribbonListButton	[In] The input ribbon list button object.
------------------	---

---

groupName	[In] The name of the group.
-----------	-----------------------------

---

---

wsTitle	[Out] It receives the title of the group.
---------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4234

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonListButtonGetItemCount****Syntax**

```
FS_INT32 FRRibbonListButtonGetItemCount (
    FR.RibbonListButton ribbonListButton,
    const FS\_CHAR\* groupName
);
```

**Description**

Get the item count.

**Parameter**

---

ribbonListButton	[In] The input ribbon list button object.
------------------	---

---

---

groupName	[In] The name of the group.
-----------	-----------------------------

---

**Return**

The item count.

**Head file reference**

fr\_barTempl.h: 4275

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonListButtonGetItemTitle****Syntax**

```
void FRRibbonListButtonGetItemTitle (
    FR.RibbonListButton ribbonListButton,
    const FS\_CHAR\* groupName,
    FS\_INT32 nItemIndex,
```

```
FS_WideString* wsTitle  
);
```

**Description**

Gets the title of the item.

**Parameter**

ribbonListButton	[In] The input ribbon list button object.
groupName	[In] The name of the group.
nItemIndex	[In] The specified index of the item.
wsTitle	[Out] It receives the title of the item.

**Return**

void

**Head file reference**

fr\_barTempl.h: 4261

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonListButtonGetItemToolTip****Syntax**

```
void FRRibbonListButtonGetItemToolTip (  
    FR_RibbonListButton ribbonListButton,  
    const FS_CHAR* groupName,  
    FS_INT32 nItemIndex,  
    FS_WideString* wsTooltip  
) ;
```

**Description**

Gets the tooltip of the item.

**Parameter**

ribbonListButton	[In] The input ribbon list button object.
groupName	[In] The name of the group.

---

nItemIndex	[In] The specified index of the item.
wsTooltip	[Out] It receives the tooltip of the item.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4247

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonListButtonGetSelectedItem****Syntax**

```
void FRRibbonListButtonGetSelectedItem (
    FR\_RibbonListButton ribbonListButton,
    FS\_ByteString* outGroupName,
    FS\_INT32* nItemIndex
);
```

**Description**

Gets the info of the selected item.

**Parameter**


---

ribbonListButton	[In] The input ribbon list button object.
outGroupName	[Out] It receives the name of the group.
nItemIndex	[Out] It receives the index of the selected item.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4287

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)

**FRRibbonListButtonIsGroupNameExist****Syntax**

```
FS_BOOL FRRibbonListButtonIsGroupNameExist (
    FR.RibbonListButton ribbonListButton,
    const FS_CHAR* name
);
```

**Description**

Checks whether the group name exists or not.

**Parameter**

---

ribbonListButton	[In] The input ribbon list button object.
------------------	---

---

name	[In] The name of the group.
------	-----------------------------

---

**Return**

[TRUE](#) if the group name exists, otherwise not.

**Head file reference**

fr\_barTempl.h: 4168

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonListButtonSetAlignedSide****Syntax**

```
void FRRibbonListButtonSetAlignedSide (
    FR.RibbonListButton ribbonListButton,
    FR.RIBBON\_LISTBUTTON\_ALIGNEDSIDE nAlignedSide
);
```

**Description**

Sets the alignment type.

**Parameter**

---

ribbonListButton	[In] The input ribbon list button object.
------------------	---

---

nAlignedSide	[In] The alignment type.
--------------	--------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4300

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonListButtonSetTitle****Syntax**

```
void FRRibbonListButtonSetTitle (
    FR\_RibbonListButton ribbonListButton,
    const FS\_CHAR* name,
    FS\_LPCWSTR lpwsGroupTitle
);
```

**Description**

Sets the title of the group.

**Parameter**


---

ribbonListButton	[In] The input ribbon list button object.
------------------	---

---

name	[In] The name of the group.
------	-----------------------------

---

lpwsGroupTitle	[Out] It receives the title of the group.
----------------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4193

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonListButtonSetIconsInRow****Syntax**

```
void FRRibbonListButtonSetIconsInRow (
    FR\_RibbonListButton ribbonListButton,
    FS\_INT32 nIconsInRow
);
```

**Description**

Sets the count of icons in a row.

**Parameter**

---

ribbonListButton	[In] The input ribbon list button object.
------------------	---

---

nIconsInRow	[In] The input count of icons in a row.
-------------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4312

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonListButtonSetTitle****Syntax**

```
void FRRibbonListButtonSetTitle (
    FR.RibbonListButton ribbonListButton,
    const FS\_CHAR* name,
    FS\_INT32 nItemIndex,
    FS\_LPCWSTR lpwsItemTitle
);
```

**Description**

Sets the title of the item.

**Parameter**

---

ribbonListButton	[In] The input ribbon list button object.
------------------	---

---

name	[In] The name of the group.
------	-----------------------------

---

nItemIndex	[In] The specified index.
------------	---------------------------

---

lpwsItemTitle	[In] The input title of the item.
---------------	-----------------------------------

---

**Return**

---

void

**Head file reference**

fr\_barTempl.h: 4220

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonListButtonSetItemToolTip****Syntax**

```
void FRRibbonListButtonSetItemToolTip (
    FR\_RibbonListButton ribbonListButton,
    const FS\_CHAR* name,
    FS\_INT32 nItemIndex,
    FS\_LPCWSTR lpwsToolTip
);
```

**Description**

Sets the tooltip of the item.

**Parameter**


---

ribbonListButton	[In] The input ribbon list button object.
------------------	---

---

name	[In] The name of the group.
------	-----------------------------

---

nItemIndex	[In] The specified index.
------------	---------------------------

---

lpwsToolTip	[In] The input tooltip of the item.
-------------	-------------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4206

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonListButtonSetSelectedItem**

**Syntax**

```
void FRRibbonListButtonSetSelectedItem (
    FR_RibbonListButton ribbonListButton,
    const FS_CHAR* csGroupName,
    FS_INT32 nItemIndex
);
```

**Description**

Sets the selected item.

**Parameter**

ribbonListButton	[In] The input ribbon list button object.
------------------	---

csGroupName	[In] The name of the group.
-------------	-----------------------------

nItemIndex	[In] The specified index.
------------	---------------------------

**Return**

void

**Head file reference**

fr\_barTempl.h: 4324

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

## FR\_RibbonPaletteButton

### [Used by](#)

**Description**

The ribbon palette button is a type of ribbon element.

**Used by**

[FRRibbonPaletteButtonAddGroup](#)  
[FRRibbonPaletteButtonEnableResize](#)  
[FRRibbonPaletteButtonGetElement](#)  
[FRRibbonPaletteButtonGetGroupCount](#)  
[FRRibbonPaletteButtonGetGroupItemCount](#)  
[FRRibbonPaletteButtonGetGroupTitle](#)  
[FRRibbonPaletteButtonGetItemData](#)  
[FRRibbonPaletteButtonGetItemInfo](#)  
[FRRibbonPaletteButtonGetItemToolTip](#)

[\*\*FRRibbonPaletteButtonGetMenuButtonTooltip\*\*](#)  
[\*\*FRRibbonPaletteButtonGetScrollButtonTooltip\*\*](#)  
[\*\*FRRibbonPaletteButtonGetSelectedItem\*\*](#)  
[\*\*FRRibbonPaletteButtonInsertItemToGroupLast\*\*](#)  
[\*\*FRRibbonPaletteButtonIsGroupNameExist\*\*](#)  
[\*\*FRRibbonPaletteButtonRemoveAllGroup\*\*](#)  
[\*\*FRRibbonPaletteButtonRemoveGroup\*\*](#)  
[\*\*FRRibbonPaletteButtonRemoveItemFromGroup\*\*](#)  
[\*\*FRRibbonPaletteButtonSetButtonMode\*\*](#)  
[\*\*FRRibbonPaletteButtonSetDefaultGroup\*\*](#)  
[\*\*FRRibbonPaletteButtonSetGroupItemEnable\*\*](#)  
[\*\*FRRibbonPaletteButtonSetGroupTitle\*\*](#)  
[\*\*FRRibbonPaletteButtonSetHighlightItemProc\*\*](#)  
[\*\*FRRibbonPaletteButtonSetIconsInRow\*\*](#)  
[\*\*FRRibbonPaletteButtonSetInitSize\*\*](#)  
[\*\*FRRibbonPaletteButtonSetItemAccNameTitle\*\*](#)  
[\*\*FRRibbonPaletteButtonSetItemData\*\*](#)  
[\*\*FRRibbonPaletteButtonSetItemToolTip\*\*](#)  
[\*\*FRRibbonPaletteButtonSetMenuButtonTooltip\*\*](#)  
[\*\*FRRibbonPaletteButtonSetRows\*\*](#)  
[\*\*FRRibbonPaletteButtonSetScrollButtonTooltip\*\*](#)  
[\*\*FRRibbonPaletteButtonSetSelectedItem\*\*](#)

## Functions

### Functions summary

#### [\*\*FRRibbonPaletteButtonAddGroup\*\*](#)

Adds a new group of ribbon palette button.

#### [\*\*FRRibbonPaletteButtonEnableResize\*\*](#)

Sets whether the palette button can be resized or not.

#### [\*\*FRRibbonPaletteButtonGetElement\*\*](#)

Gets the ribbon element associated with the ribbon palette button. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

#### [\*\*FRRibbonPaletteButtonGetGroupCount\*\*](#)

Gets the group count.

#### [\*\*FRRibbonPaletteButtonGetGroupItemCount\*\*](#)

Gets the item count of the group.

#### [\*\*FRRibbonPaletteButtonGetGroupTitle\*\*](#)

Gets the title of the group.

#### [\*\*FRRibbonPaletteButtonGetData\*\*](#)

Gets the item data.

#### [\*\*FRRibbonPaletteButtonGetItemInfo\*\*](#)

Gets the item info in the group.

#### [\*\*FRRibbonPaletteButtonGetItemToolTip\*\*](#)

Gets the tooltip of the item.

#### [\*\*FRRibbonPaletteButtonGetMenuButtonTooltip\*\*](#)

Gets the tooltip of the drop-down button on the palette.

#### [\*\*FRRibbonPaletteButtonGetScrollButtonTooltip\*\*](#)

Gets the tooltip of the scroll button.

#### [\*\*FRRibbonPaletteButtonGetSelectedItem\*\*](#)

Gets the info of the selected item.

**FRRibbonPaletteButtonInsertItemToGroupLast**

Inserts a new item to the existing group. If the group does not exist, it will be created.

**FRRibbonPaletteButtonIsGroupNameExist**

Checks whether the name of the group exists or not.

**FRRibbonPaletteButtonRemoveAllGroup**

Removes all the groups.

**FRRibbonPaletteButtonRemoveGroup**

Removes the specified group.

**FRRibbonPaletteButtonRemoveItemFromGroup**

Removes the specified item from the group.

**FRRibbonPaletteButtonSetButtonMode**

Whether to set the drop-down button mode or not.

**FRRibbonPaletteButtonSetDefaultGroup**

Sets the default group.

**FRRibbonPaletteButtonSetGroupItemEnable**

Enables the item or not.

**FRRibbonPaletteButtonSetGroupTitle**

Sets the title of the group.

**FRRibbonPaletteButtonSetHighlightItemProc**

Sets the callback invoked when the item is highlight.

**FRRibbonPaletteButtonSetIconsInRow**

Sets the count of icons in a row.

**FRRibbonPaletteButtonSetInitSize**

Sets the init size.

**FRRibbonPaletteButtonSetItemAccNameTitle**

Sets the callback invoked when the item is highlight.

**FRRibbonPaletteButtonSetItemData**

Sets the item data.

**FRRibbonPaletteButtonSetItemToolTip**

Sets the tooltip of the item.

**FRRibbonPaletteButtonSetMenuButtonTooltip**

Sets the tooltip of the drop-down button on the palette.

**FRRibbonPaletteButtonSetRows**

Sets the rows.

**FRRibbonPaletteButtonSetScrollButtonTooltip**

Sets the tooltip of the scroll button.

**FRRibbonPaletteButtonSetSelectedItem**

Sets the selected item.

## Functions detail

### FRRibbonPaletteButtonAddGroup

#### Syntax

```
void FRRibbonPaletteButtonAddGroup (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    const FS\_CHAR* name,
    FS\_LPCWSTR lpwsGroupTitle,
    FS\_DIBitmap paletteImages,
    FS\_INT32 nWidthForEachImage
);
```

**Description**

Adds a new group of ribbon palette button.

**Parameter**


---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

name	[In] The input name of the group.
------	-----------------------------------

---

lpwsGroupTitle	[In] The input title of the group.
----------------	------------------------------------

---

paletteImages	[In] The input bitmap list of the group.
---------------	--

---

nWidthForEachImage	[In] The width of each bitmap.
--------------------	--------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3307

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonPaletteButtonEnableResize****Syntax**

```
void FRRibbonPaletteButtonEnableResize (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    FS\_BOOL bEnable,
    FS\_BOOL bVerticalOnly
);
```

**Description**

Sets whether the palette button can be resized or not.

**Parameter**


---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

bEnable	[In] TRUE if the palette button can be resized.
---------	---

---

bVerticalOnly	[In] TRUE if the palette button can be resized in vertical direction only.
---------------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3483

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonPaletteButtonGetElement****Syntax**

```
FR_RibbonElement FRRibbonPaletteButtonGetElement (
    FR\_RibbonPaletteButton ribbonPaletteButton
);
```

**Description**

Gets the ribbon element associated with the ribbon palette button. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**Parameter**

ribbonPaletteButton	[In]
---------------------	------

---

**Return**

The ribbon element associated with the ribbon palette button.

**Head file reference**

fr\_barTempl.h: 3295

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonPaletteButtonGetGroupCount****Syntax**

```
FS_INT32 FRRibbonPaletteButtonGetGroupCount (
    FR\_RibbonPaletteButton ribbonPaletteButton
);
```

**Description**

Gets the group count.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

**Return**

The group count.

**Head file reference**

fr\_barTempl.h: 3436

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonPaletteButtonGetGroupItemCount****Syntax**

```
FS_INT32 FRRibbonPaletteButtonGetGroupItemCount (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    const FS_CHAR* groupName
);
```

**Description**

Gets the item count of the group.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

groupName	[In] The input name of the group.
-----------	-----------------------------------

---

**Return**

The item count of the group.

**Head file reference**

fr\_barTempl.h: 3424

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonPaletteButtonGetGroupTitle**

**Syntax**

```
void FRRibbonPaletteButtonGetGroupTitle (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    const FS\_CHAR\* groupName,
    FS\_WideString\* wsTitle
);
```

**Description**

Gets the title of the group.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

groupName	[In] The input name of the group.
-----------	-----------------------------------

---

wsTitle	[Out] It receives the title of the group.
---------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3373

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonPaletteButtonGetData****Syntax**

```
FS_DWORD FRRibbonPaletteButtonGetData (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    const FS\_CHAR\* groupName,
    FS\_INT32 nIndex
);
```

**Description**

Gets the item data.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

groupName	[In] The specified group name.
-----------	--------------------------------

---

---

nIndex	[In] The specified index of the item.
--------	---------------------------------------

---

**Return**

The item data.

**Head file reference**

fr\_barTempl.h: 3576

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonPaletteButtonGetItemInfo****Syntax**

```
void FRRibbonPaletteButtonGetItemInfo (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    FS\_INT32 nIndex,
    FS\_BytString* outGroupName,
    FS\_INT32* outItemIndex
);
```

**Description**

Gets the item info in the group.

**Parameter**


---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

nIndex	[In] The input index of the the item.
--------	---------------------------------------

---

outGroupName	[Out] It receives the name of the group that the item belongs to.
--------------	---

---

outItemIndex	[Out] It receives the index in the group that the item belongs to.
--------------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3652

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

**FRRibbonPaletteButtonGetItemToolTip****Syntax**

```
void FRRibbonPaletteButtonGetItemToolTip (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    const FS\_CHAR* groupName,
    FS\_INT32 nItemIndex,
    FS\_WideString* wsTooltip
);
```

**Description**

Gets the tooltip of the item.

**Parameter**

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

groupName	[In] The input name of the group.
-----------	-----------------------------------

nItemIndex	[In] The specified index of the item.
------------	---------------------------------------

wsTooltip	[Out] It receives the tooltip of the item.
-----------	--

**Return**

void

**Head file reference**

[fr\\_barTempl.h](#): 3386

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonPaletteButtonGetMenuItemTooltip****Syntax**

```
void FRRibbonPaletteButtonGetMenuItemTooltip (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    FS\_WideString* wsTooltip
);
```

**Description**

Gets the tooltip of the drop-down button on the palette.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

wsTooltip	[Out] It receives the tooltip of the drop-down button on the palette.
-----------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3412

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonPaletteButtonGetScrollButtonTooltip****Syntax**

```
void FRRibbonPaletteButtonGetScrollButtonTooltip (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    FS\_WideString* wsTooltip
);
```

**Description**

Gets the tooltip of the scroll button.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

wsTooltip	[Out] It receives the tooltip of the scroll button.
-----------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3400

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonPaletteButtonGetSelectedItem**

**Syntax**

```
void FRRibbonPaletteButtonGetSelectedItem (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    FS\_BytString* outGroupName,
    FS\_INT32* nItemIndex
);
```

**Description**

Gets the info of the selected item.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

outGroupName	[Out] It receives the group name of the selected item.
--------------	--

---

nItemIndex	[Out] It receives the index of the selected item.
------------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3496

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

[FRRibbonPaletteButtonInsertItemToGroupLast](#)

**Syntax**

```
void FRRibbonPaletteButtonInsertItemToGroupLast (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    const FS\_CHAR* groupName,
    FS\_LPCWSTR lpwsGroupTitle,
    FS\_DIBitmap pBitmap,
    FS\_INT32 cxPaletteImage
);
```

**Description**

Inserts a new item to the existing group. If the group does not exist, it will be created.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

---

groupName	[In] The specified group name.
lpwsGroupTitle	[In] The input group title if the group name does not exist.
pBitmap	[In] The input bitmap list of the group.
cxPaletteImage	[In] The width of each bitmap.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3509

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonPaletteButtonIsGroupNameExist****Syntax**

```
FS_BOOL FRRibbonPaletteButtonIsGroupNameExist (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    const FS\_CHAR\* groupName
);
```

**Description**

Checks whether the name of the group exists or not.

**Parameter**


---

ribbonPaletteButton	[In] The input ribbon palette button object.
groupName	[In] The input name of the group.

---

**Return**[TRUE](#) if the name of the group exists, otherwise not.**Head file reference**

fr\_barTempl.h: 3447

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)

## FRRibbonPaletteButtonRemoveAllGroup

### Syntax

```
void FRRibbonPaletteButtonRemoveAllGroup (
    FR\_RibbonPaletteButton ribbonPaletteButton
);
```

### Description

Removes all the groups.

### Parameter

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

### Return

void

### Head file reference

fr\_barTempl.h: 3628

### Related method

#### Since

[SDK LATEEST VERSION > 2.1.0.4](#)

## FRRibbonPaletteButtonRemoveGroup

### Syntax

```
void FRRibbonPaletteButtonRemoveGroup (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    const FS\_CHAR* groupName
);
```

### Description

Removes the specified group.

### Parameter

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

groupName	[In] The specified name.
-----------	--------------------------

---

### Return

void

**Head file reference**

fr\_barTempl.h: 3537

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonPaletteButtonRemoveItemFromGroup****Syntax**

```
void FRRibbonPaletteButtonRemoveItemFromGroup (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    const FS\_CHAR\* groupName,
    FS\_INT32 nIndex
);
```

**Description**

Removes the specified item from the group.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

groupName	[In] The specified group name.
-----------	--------------------------------

---

nIndex	[In] The specified index of the item.
--------	---------------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3549

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonPaletteButtonSetButtonMode****Syntax**

```
void FRRibbonPaletteButtonSetButtonMode (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    FS\_BOOL bDropDown
);
```

**Description**

Whether to set the drop-down button mode or not.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

bDropDown	[In] TRUE if you want to set the drop-down button mode.
-----------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3459

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonPaletteButtonSetDefaultGroup****Syntax**

```
void FRRibbonPaletteButtonSetDefaultGroup (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    FS\_DIBitmap pBitmap,
    FS\_INT32 cxPaletteImage
);
```

**Description**

Sets the default group.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

pBitmap	[In] The input bitmap list of the default group.
---------	--

---

cxPaletteImage	[In] The width of each bitmap.
----------------	--------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3524

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonPaletteButtonSetGroupItemEnable****Syntax**

```
void FRRibbonPaletteButtonSetGroupItemEnable (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    const FS\_CHAR\* groupName,
    FS\_INT32 nIndex,
    FS\_BOOL bEnable
);
```

**Description**

Enables the item or not.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

groupName	[In] The specified group name.
-----------	--------------------------------

---

nIndex	[In] The specified index of the item.
--------	---------------------------------------

---

bEnable	[In] Sets TRUE to enable the item, otherwise FALSE.
---------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3589

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonPaletteButtonSetGroupTitle****Syntax**

```
void FRRibbonPaletteButtonSetGroupTitle (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    const FS\_CHAR\* name,
    FS\_LPCWSTR lpwsGroupTitle
```

);

**Description**

Sets the title of the group.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

name	[In] The input name of the group.
------	-----------------------------------

---

lpwsGroupTitle	[In] The input title of the group.
----------------	------------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3322

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonPaletteButtonSetHighlightItemProc****Syntax**

```
void FRRibbonPaletteButtonSetHighlightItemProc (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    FRRibbonPaletteButtonHighlightItemProc proc
);
```

**Description**

Sets the callback invoked when the item is highlight.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

proc	[In] The callback invoked when the item is highlight.
------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3666

#### Related method

##### Since

[SDK LATEEST VERSION > 7.1.0.0](#)

FRRibbonPaletteButtonSetIconsInRow

#### Syntax

```
void FRRibbonPaletteButtonSetIconsInRow (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    FS\_INT32 nIconsInRow
);
```

#### Description

Sets the count of icons in a row.

#### Parameter

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

nIconsInRow	[In] The input count of icons in a row.
-------------	---

---

#### Return

void

#### Head file reference

fr\_barTempl.h: 3471

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

FRRibbonPaletteButtonSetInitSize

#### Syntax

```
void FRRibbonPaletteButtonSetInitSize (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    FS\_INT32 cx,
    FS\_INT32 cy
);
```

#### Description

Sets the init size.

**Parameter**


---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

cx	[In] The width in x-coordinate.
----	---------------------------------

---

cy	[In] The height in y-coordinate.
----	----------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3639

**Related method****Since**[SDK LATEEST VERSION > 2.1.0.4](#)**FRRibbonPaletteButtonSetItemAccNameTitle****Syntax**

```
void FRRibbonPaletteButtonSetItemAccNameTitle (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    FS\_ByteString bsGroupName,
    FS\_INT32 nIndex,
    FS\_WideString wsAccName
);
```

**Description**

Sets the callback invoked when the item is highlight.

**Parameter**


---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

bsGroupName	[In] The name of the group that the item belongs to.
-------------	--

---

nIndex	[In] The index in the group that the item belongs to.
--------	---

---

wsAccName	[In] The name provided to the Third party reading tool
-----------	--

---

**Return**

void

**Head file reference**

---

fr\_barTempl.h: 3678

#### Related method

##### Since

[SDK LATEEST VERSION > 9.1.0.0](#)

## FRRibbonPaletteButtonSetItemData

#### Syntax

```
void FRRibbonPaletteButtonSetItemData (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    const FS\_CHAR\* groupName,
    FS\_INT32 nIndex,
    FS\_DWORD dwData
);
```

#### Description

Sets the item data.

#### Parameter

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

groupName	[In] The specified group name.
-----------	--------------------------------

---

nIndex	[In] The specified index of the item.
--------	---------------------------------------

---

dwData	[In] The input item data.
--------	---------------------------

---

#### Return

void

#### Head file reference

fr\_barTempl.h: 3562

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

## FRRibbonPaletteButtonSetItemToolTip

#### Syntax

```
void FRRibbonPaletteButtonSetItemToolTip (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    const FS\_CHAR\* name,
```

```
FS_INT32 nItemIndex,  
FS_LPCWSTR lpwsToolTip  
);
```

**Description**

Sets the tooltip of the item.

**Parameter**

ribbonPaletteButton	[In] The input ribbon palette button object.
name	[In] The input name of the group.
nItemIndex	[In] The specified index of the item.
lpwsToolTip	[In] The input tooltip of the item.

**Return**

void

**Head file reference**

fr\_barTempl.h: 3335

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonPaletteButtonSetMenuButtonTooltip****Syntax**

```
void FRRibbonPaletteButtonSetMenuButtonTooltip (  
    FR_RibbonPaletteButton ribbonPaletteButton,  
    FS_LPCWSTR lpwsTooltip  
);
```

**Description**

Sets the tooltip of the drop-down button on the palette.

**Parameter**

ribbonPaletteButton	[In] The input ribbon palette button object.
lpwsTooltip	[In] The input tooltip of the drop-down button on the palette.

**Return**

void

**Head file reference**

fr\_barTempl.h: 3361

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonPaletteButtonSetRows****Syntax**

```
void FRRibbonPaletteButtonSetRows (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    FS\_INT32 nRows
);
```

**Description**

Sets the rows.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

nRows	[In] The input rows.
-------	----------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3616

**Related method****Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FRRibbonPaletteButtonSetScrollButtonTooltip****Syntax**

```
void FRRibbonPaletteButtonSetScrollButtonTooltip (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    FS\_LPCWSTR lpwsTooltip
);
```

**Description**

Sets the tooltip of the scroll button.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

lpwsTooltip	[In] The input tooltip of the scroll button.
-------------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3349

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonPaletteButtonSetSelectedItem****Syntax**

```
void FRRibbonPaletteButtonSetSelectedItem (
    FR\_RibbonPaletteButton ribbonPaletteButton,
    const FS\_CHAR\* csGroupName,
    FS\_INT32 nItemIndex
);
```

**Description**

Sets the selected item.

**Parameter**

---

ribbonPaletteButton	[In] The input ribbon palette button object.
---------------------	--

---

csGroupName	[In] The specified group name.
-------------	--------------------------------

---

nItemIndex	[In] The specified index of the item.
------------	---------------------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 3603

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)

## FR\_RibbonPanel

[\*\*Return from Used by\*\*](#)

### Description

The ribbon panel can be used to classify the operation. You can put the ribbon element on the ribbon panel. The ribbon panel object s managed by the ribbon category. You can add a ribbon panel through [FRRibbonCategoryAddPanel](#) .

### Returned from

[FRRibbonCategoryAddPanel](#)  
[FRRibbonCategoryAddPanel2](#)  
[FRRibbonCategoryCopyPanel](#)  
[FRRibbonCategoryGetPanelByIndex](#)  
[FRRibbonCategoryGetPanelByName](#)

### Used by

[FRRibbonCategoryCopyPanel](#)  
[FRRibbonPanelAddDialog](#)  
[FRRibbonPanelAddElement](#)  
[FRRibbonPanelAddElementToGroup](#)  
[FRRibbonPanelChangeElementType](#)  
[FRRibbonPanelCopyElementToPanel](#)  
[FRRibbonPanelEnableAddToCustomizeDlg](#)  
[FRRibbonPanelGetElementByIndex](#)  
[FRRibbonPanelGetElementByName](#)  
[FRRibbonPanelGetElementCount](#)  
[FRRibbonPanelGetName](#)  
[FRRibbonPanelGetTitle](#)  
[FRRibbonPanelGetVisible](#)  
[FRRibbonPanelPreTranslateMessage](#)  
[FRRibbonPanelSetImageInitProc](#)  
[FRRibbonPanelSetKey](#)  
[FRRibbonPanelSetLaunchButton](#)  
[FRRibbonPanelSetPanelImage](#)  
[FRRibbonPanelSetShowDefaultButtonAtLast](#)  
[FRRibbonPanelSetTitle](#)  
[FRRibbonPanelSetVisible](#)  
[FRRibbonPanelShowInQATCustomizeToolsDlg](#)

### Callbacks

#### Callbacks summary

**FRPanelDlgCreateProc**

The callback function is called to notify the plug-in to create the dialog attached to the panel.

**FRPanelDlgDestoryProc**

The callback function is called to notify the plug-in to destroy the dialog attached to the panel.

**Callbacks detail****FRPanelDlgCreateProc****Syntax**

```
typedef HWND (*FRPanelDlgCreateProc)(  
    HWND hParentWnd,  
    void* pDialog  
)
```

**Description**

The callback function is called to notify the plug-in to create the dialog attached to the panel.

**Parameter**


---

hParentWnd	[In] The parent window handle.
pDialog	[In] The pointer to a <i>MFC CDialog</i> attached to the panel. It is passed from <a href="#">FRRibbonPanelAddDialog</a> .

---

**Return**

The dialog handle created.

**Head file reference**

fr\_barExpT.h: 394

**Related method**

[FRRibbonPanelAddDialog](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRPanelDlgDestoryProc****Syntax**

```
typedef void (*FRPanelDlgDestoryProc)(  
    void* pDialog  
)
```

**Description**

The callback function is called to notify the plug-in to destroy the dialog attached to the panel.

### Parameter

---

pDialog	[In] The dialog handle to be destroyed. It is passed from <a href="#">FRRibbonPanelAddDialog</a> .
---------	--

---

### Return

void

### Head file reference

fr\_barExpT.h: 406

### Related method

[FRRibbonPanelAddDialog](#)

### Since

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

## Functions

### Functions summary

#### [FRRibbonPanelAddDialog](#)

You can create your own dialog under the panel. You have to invoke [FRRibbonPanelPreTranslateMessage](#) in the dialog's *PreTranslateMessage* function to dispatch the message.

#### [FRRibbonPanelAddElement](#)

Adds a new ribbon element to the ribbon panel

#### [FRRibbonPanelAddElementToGroup](#)

Adds a new ribbon element to the specified group.

#### [FRRibbonPanelChangeElementType](#)

Changes the specified element to other types.

#### [FRRibbonPanelCopyElementToPanel](#)

Copies an existing ribbon element to the panel.

#### [FRRibbonPanelEnableAddToCustomizeDlg](#)

Sets whether the panel can be added to the custom setting dialog or not.

#### [FRRibbonPanelGetElementByIndex](#)

Gets the the specified ribbon element by index.

#### [FRRibbonPanelGetElementByName](#)

Gets the The specified ribbon element by name.

#### [FRRibbonPanelGetElementCount](#)

Get the count of the ribbon element.

#### [FRRibbonPanelGetName](#)

Gets the name of the ribbon panel.

#### [FRRibbonPanelGetTitle](#)

Gets the title of the ribbon panel.

#### [FRRibbonPanelGetVisible](#)

CHhecks whether the ribbon panel is visible or not.

#### **FRRibbonPanelPreTranslateMessage**

When you create a dialog under the panel, you have to invoke **FRRibbonPanelPreTranslateMessage** in the dialog's *PreTranslateMessage* function to dispatch the message.

#### **FRRibbonPanelSetImageInitProc**

Sets the callback invoked by Foxit Reader to init the visible image.

#### **FRRibbonPanelSetKey**

Sets the shortcut key of the ribbon panel

#### **FRRibbonPanelSetLaunchButton**

Sets the launch button.

#### **FRRibbonPanelSetPanelImage**

Sets the bitmap when the ribbon panel is narrowed.

#### **FRRibbonPanelSetShowDefaultButtonAtLast**

Sets whether the panel will show the default button at last or not.

#### **FRRibbonPanelSetTitle**

Sets the title of the ribbon panel.

#### **FRRibbonPanelSetVisible**

Sets the ribbon panel to visible or not.

#### **FRRibbonPanelShowInQATCustomizeToolsDlg**

Sets whether the panel can be shown in the QAT customize tools dialog.

## Functions detail

### **FRRibbonPanelAddDialog**

#### Syntax

```
FS_BOOL FRRibbonPanelAddDialog (
    FR_RibbonPanel ribbonPanel,
    FRPanelDlgCreateProc createProc,
    FRPanelDlgDestoryProc destroyProc,
    void* pDialog
);
```

#### Description

You can create your own dialog under the panel. You have to invoke **FRRibbonPanelPreTranslateMessage** in the dialog's *PreTranslateMessage* function to dispatch the message.

#### Parameter

ribbonPanel	[In] The input ribbon panel object.
createProc	[In] The callback function is called to notify the plug-in to create the dialog attached to the panel.
destroyProc	[In] The callback function is called to notify the plug-in to destroy the dialog attached to the panel.

---

pDialog	[In] The input pointer to a <i>MFC CDialog</i> attached to the panel. It will be passed to the <i>createProc</i> and <i>destroyProc</i> .
---------	--

---

**Return**TRUE for success, otherwise not.**Head file reference**

fr\_barTempl.h: 1857

**Related method****Since**SDK\_LATEEST\_VERSION > 7.2.2**FRRibbonPanelAddElement****Syntax**

```
void* FRRibbonPanelAddElement (
    FR\_RibbonPanel ribbonPanel,
    FR.Ribbon\_Element\_Type nElementType,
    const FS\_CHAR\* name,
    FS\_LPCWSTR lpwsTitle,
    FS\_INT32 nPos
);
```

**Description**

Adds a new ribbon element to the ribbon panel

**Parameter**


---

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

---

nElementType	[In] The specified type of the element to be added.
--------------	---

---

name	[In] The specified name of the element to be added.
------	---

---

lpwsTitle	[In] The specified title of the element to be added.
-----------	--

---

nPos	[In] The specified position of the element to be added.
------	---

---

**Return**The new ribbon element. For example, if sets *nElementType* as [FR\\_RIBBON\\_BUTTON](#) , the returned value can be converted to [FR.RibbonButton](#) .**Head file reference**

fr\_barTempl.h: 1644

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonPanelAddElementToGroup****Syntax**

```
void* FRRibbonPanelAddElementToGroup (
    FR\_RibbonPanel ribbonPanel,
    const FS_CHAR* groupName,
    FR.Ribbon\_Element\_Type nElementType,
    const FS_CHAR* elementName,
    FS\_LPCWSTR lpwsTitle
);
```

**Description**

Adds a new ribbon element to the specified group.

**Parameter**

ribbonPanel	[In] The input ribbon panel object.
groupName	[In] The specified group name.
nElementType	[In] The type of ribbon element added to the group.
elementName	[In] The input name of the ribbon element.
lpwsTitle	[In] The input title of the ribbon element.

**Return**

The new ribbon element added to the specified group. For example, if sets *elementName* as [FR\\_RIBBON\\_BUTTON](#), the returned value can be converted to [FR.RibbonButton](#). The element belongs to the group will not display the title.

**Head file reference**

fr\_barTempl.h: 1673

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonPanelChangeElementType**

**Syntax**

```
void* FRRibbonPanelChangeElementType (
    FR\_RibbonPanel ribbonPanel,
    const FS\_CHAR\* name,
    FR\_Ribbon\_Element\_Type nElementType
);
```

**Description**

Changes the specified element to other types.

**Parameter**

---

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

---

name	[In] The specified name of the ribbon element.
------	--

---

nElementType	[In] The specified type you want to change to.
--------------	--

---

**Return**

The ribbon element with new type.

**Head file reference**

fr\_barTempl.h: 1660

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

[FRRibbonPanelCopyElementToPanel](#)

**Syntax**

```
void* FRRibbonPanelCopyElementToPanel (
    FR\_RibbonPanel ribbonPanel,
    FR\_RibbonElement pSrcRibbonElement
);
```

**Description**

Copies an existing ribbon element to the panel.

**Parameter**

---

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

---

pSrcRibbonElement	[In] The existing ribbon element.
-------------------	-----------------------------------

---

**Return**

The corresponding ribbon button. For example, if the type is [FR\\_RIBBON\\_BUTTON](#), the returned value can be converted to [FR.RibbonButton](#).

**Head file reference**

fr\_barTempl.h: 1702

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonPanelEnableAddToCustomizeDlg****Syntax**

```
void FRRibbonPanelEnableAddToCustomizeDlg (
    FR.RibbonPanel ribbonPanel,
    FS\_BOOL bEnable
);
```

**Description**

Sets whether the panel can be added to the custom setting dialog or not.

**Parameter**

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

bEnable	[In] Whether the panel can be added to the custom setting dialog or not.
---------	--

**Return**

void

**Head file reference**

fr\_barTempl.h: 1885

**Since**

[SDK LATEEST VERSION > 7.2.2](#)

**FRRibbonPanelGetElementByIndex****Syntax**

```
FR_RibbonElement FRRibbonPanelGetElementByIndex (
    FR.RibbonPanel ribbonPanel,
    FS\_INT32 nIndex
);
```

**Description**

Gets the the specified ribbon element by index.

**Parameter**

---

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

---

nIndex	[In] The specified index.
--------	---------------------------

---

**Return**

The specified ribbon element.

**Head file reference**

fr\_barTempl.h: 1785

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonPanelGetElementByName****Syntax**

```
FR_RibbonElement FRRibbonPanelGetElementByName (
    FR_RibbonPanel ribbonPanel,
    const FS_CHAR* elementName
);
```

**Description**

Gets the The specified ribbon element by name.

**Parameter**

---

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

---

elementName	[In] The specified name of the ribbon element.
-------------	--

---

**Return**

The specified ribbon element.

**Head file reference**

fr\_barTempl.h: 1762

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonPanelGetElementCount

### Syntax

```
FS_INT32 FRRibbonPanelGetElementCount (
    FR\_RibbonPanel ribbonPanel
);
```

### Description

Get the count of the ribbon element.

### Parameter

---

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

---

### Return

The count of the ribbon element.

### Head file reference

fr\_barTempl.h: 1774

### Related method

#### Since

[SDK LATEEST VERSION > 2.0](#)

## FRRibbonPanelGetName

### Syntax

```
void FRRibbonPanelGetName (
    FR\_RibbonPanel ribbonPanel,
    FS\_ByteString* bsName
);
```

### Description

Gets the name of the ribbon panel.

### Parameter

---

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

---

bsName	[Out] It receives the name of the ribbon panel.
--------	---

---

### Return

void

**Head file reference**

fr\_barTempl.h: 1809

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonPanelGetTitle****Syntax**

```
void FRRibbonPanelGetTitle (
    FR\_RibbonPanel ribbonPanel,
    FS\_WideString* wsTitle
);
```

**Description**

Gets the title of the ribbon panel.

**Parameter**

---

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

---

wsTitle	[Out] It receives the title of the ribbon panel.
---------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1727

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonPanelGetVisible****Syntax**

```
FS_BOOL FRRibbonPanelGetVisible (
    FR\_RibbonPanel ribbonPanel
);
```

**Description**

CHecks whether the ribbon panel is visible or not.

**Parameter**

---

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

---

**Return**

[TRUE](#) if the ribbon panel is visible, otherwise [FALSE](#).

**Head file reference**

fr\_barTempl.h: 1751

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonPanelPreTranslateMessage****Syntax**

```
FS_BOOL FRRibbonPanelPreTranslateMessage (
    FR.RibbonPanel ribbonPanel,
    void* pMsg
);
```

**Description**

When you create a dialog under the panel, you have to invoke [FRRibbonPanelPreTranslateMessage](#) in the dialog's *PreTranslateMessage* function to dispatch the message.

**Parameter**


---

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

---

pMsg	[In] Pointer to a <i>MFC MSG</i> structure that contains the message to process.
------	--

---

**Return**

[TRUE](#) if the message was fully processed and should not be processed further. [FALSE](#) if the message should be processed in the normal way.

**Head file reference**

fr\_barTempl.h: 1858

**Related method**

[FRRibbonPanelAddDialog](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRRibbonPanelSetImageInitProc**

**Syntax**

```
void FRRibbonPanelSetImageInitProc (
    FR\_RibbonPanel ribbonPanel,
    FRRibbonPanelImageInitProc proc
);
```

**Description**

Sets the callback invoked by Foxit Reader to init the visible image.

**Parameter**

---

ribbonPanel	[In] The input ribbon panel object.
proc	[In] The callback invoked by Foxit Reader to init the visible image.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1833

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FRRibbonPanelSetKey****Syntax**

```
void FRRibbonPanelSetKey (
    FR\_RibbonPanel ribbonPanel,
    FS\_LPCWSTR lpwsShorcutKey
);
```

**Description**

Sets the shortcut key of the ribbon panel

**Parameter**

---

ribbonPanel	[In] The input ribbon panel object.
lpwsShorcutKey	[In] The input shortcut key.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1845

**Related method****Since**[SDK LATEEST VERSION > 3.0.0.0](#)**FRRibbonPanelSetLaunchButton****Syntax**

```
FR_RibbonButton FRRibbonPanelSetLaunchButton (
    FR.RibbonPanel ribbonPanel,
    const FS\_CHAR\* name,
    FS\_LPCWSTR lpwsTitle
);
```

**Description**

Sets the launch button.

**Parameter**

---

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

---

name	[In] The input name of the launch button.
------	---

---

lpwsTitle	[In] The input title of the launch button.
-----------	--

---

**Return**

The launch button.

**Head file reference**

fr\_barTempl.h: 1689

**Related method****Note:** This interface is reserved.**Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonPanelSetPanelImage****Syntax**

```
void FRRibbonPanelSetPanelImage (
    FR.RibbonPanel ribbonPanel,
```

---

```
FS_DIBitmap pSmallBitmap
);
```

**Description**

Sets the bitmap when the ribbon panel is narrowed.

**Parameter**


---

ribbonPanel	[In] The input ribbon panel object.
pSmallBitmap	[In] The input bitmap when the ribbon panel is narrowed.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1797

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonPanelSetShowDefaultButtonAtLast****Syntax**

```
void FRRibbonPanelSetShowDefaultButtonAtLast (
    FR_RibbonPanel ribbonPanel,
    FS_BOOL bLast
);
```

**Description**

Sets whether the panel will show the default button at last or not.

**Parameter**


---

ribbonPanel	[In] The input ribbon panel object.
bLast	[In] Whether the panel will show the default button at last or not.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1896

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRRibbonPanelSetTitle****Syntax**

```
void FRRibbonPanelSetTitle (
    FR\_RibbonPanel ribbonPanel,
    FS\_LPCWSTR lpwsTitle
);
```

**Description**

Sets the title of the ribbon panel.

**Parameter**

---

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

---

lpwsTitle	[In] The input title of the ribbon panel.
-----------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1715

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonPanelSetVisible****Syntax**

```
void FRRibbonPanelSetVisible (
    FR\_RibbonPanel ribbonPanel,
    FS\_BOOL bVisible
);
```

**Description**

Sets the ribbon panel to visible or not.

**Parameter**

---

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

---

---

bVisible	[In] <a href="#">TRUE</a> if sets the ribbon panel to visible.
----------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1739

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonPanelShowInQATCustomizeToolsDlg****Syntax**

```
void FRRibbonPanelShowInQATCustomizeToolsDlg (
    FR.RibbonPanel ribbonPanel,
    FS\_BOOL bShow
);
```

**Description**

Sets whether the panel can be shown in the QAT customize tools dialog.

**Parameter**


---

ribbonPanel	[In] The input ribbon panel object.
-------------	-------------------------------------

---

bShow	[In] It indicates whether the panel can be shown in the QAT customize tools dialog.
-------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 1821

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)**FR\_RibbonRadioButton****[Used by](#)**

## Description

The ribbon radio button is a type of ribbon element.

## Used by

[\*\*FRRibbonRadioButtonGetElement\*\*](#)  
[\*\*FRRibbonRadioButtonIsChecked\*\*](#)  
[\*\*FRRibbonRadioButtonSetCheck\*\*](#)

## Functions

### Functions summary

#### [\*\*FRRibbonRadioButtonGetElement\*\*](#)

Gets the ribbon element associated with the ribbon radio button. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

#### [\*\*FRRibbonRadioButtonIsChecked\*\*](#)

Checks whether the ribbon radio button is checked or not.

#### [\*\*FRRibbonRadioButtonSetCheck\*\*](#)

Whether to set the ribbon radio button to be checked or not.

### Functions detail

#### FRRibbonRadioButtonGetElement

##### Syntax

```
FR_RibbonElement FRRibbonRadioButtonGetElement (
    FR\_RibbonRadioButton ribbonRadioButton
);
```

##### Description

Gets the ribbon element associated with the ribbon radio button. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

##### Parameter

---

ribbonRadioButton	[In] The input ribbon radio button object.
-------------------	--

---

##### Return

The ribbon element associated with the ribbon radio button.

##### Head file reference

fr\_barTempl.h: 2778

##### Related method

##### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonRadioButtonIsChecked

### Syntax

```
FS_BOOL FRRibbonRadioButtonIsChecked (
    FR\_RibbonRadioButton ribbonRadioButton
);
```

### Description

Checks whether the ribbon radio button is checked or not.

### Parameter

---

ribbonRadioButton	[In] The input ribbon radio button object.
-------------------	--

---

### Return

[TRUE](#) if the ribbon radio button is checked, otherwise not.

### Head file reference

fr\_barTempl.h: 2790

### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonRadioButtonSetCheck

### Syntax

```
void FRRibbonRadioButtonSetCheck (
    FR\_RibbonRadioButton ribbonRadioButton,
    FS_BOOL bCheck
);
```

### Description

Whether to set the ribbon radio button to be checked or not.

### Parameter

---

ribbonRadioButton	[In] The input ribbon radio button object.
-------------------	--

---

bCheck	[In] <a href="#">TRUE</a> if you want to set the ribbon radio button to be checked.
--------	---

---

### Return

void

### Head file reference

fr\_barTempl.h: 2801

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

## FR\_RibbonSlider

### [Used by](#)

#### Description

The ribbon slider is a type of ribbon element.

#### Used by

[FRRibbonSliderGetElement](#)  
[FRRibbonSliderGetMouseStatus](#)  
[FRRibbonSliderGetPos](#)  
[FRRibbonSliderGetRangeMax](#)  
[FRRibbonSliderGetRangeMin](#)  
[FRRibbonSliderSetPos](#)  
[FRRibbonSliderSetRange](#)  
[FRRibbonSliderSetStyle](#)  
[FRRibbonSliderSetWidth](#)  
[FRRibbonSliderSetZoomButtons](#)

#### Functions

##### Functions summary

###### [FRRibbonSliderGetElement](#)

Gets the ribbon element associated with the ribbon slider. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

###### [FRRibbonSliderGetMouseStatus](#)

Gets the mouse status. -1 - none, 0 - LButtonDown, 1 - LButtonUp, 2 - LButtonDown & MouseMove.

###### [FRRibbonSliderGetPos](#)

Gets the position of the slider.

###### [FRRibbonSliderGetRangeMax](#)

Gets the maximum value of the range.

###### [FRRibbonSliderGetRangeMin](#)

Gets the minimum value of the range.

###### [FRRibbonSliderSetPos](#)

Sets the position of the slider.

###### [FRRibbonSliderSetRange](#)

Sets the range of the slider.

###### [FRRibbonSliderSetStyle](#)

Sets the style of the ribbon slider.

**FRRibbonSliderSetWidth**

Sets the width of the ribbon slider.

**FRRibbonSliderSetZoomButtons**

Sets whether to add the zoom buttons or not.

## Functions detail

### FRRibbonSliderGetElement

**Syntax**

```
FR_RibbonElement FRRibbonSliderGetElement (
    FR\_RibbonSlider ribbonSlider
);
```

**Description**

Gets the ribbon element associated with the ribbon slider. All types of buttons associate with a ribbon element. The ribbon element manipulates the common properties of all types of buttons.

**Parameter**

---

ribbonSlider	[In] The input ribbon slider object.
--------------	--------------------------------------

---

**Return**

The ribbon element associated with the ribbon slider.

**Head file reference**

fr\_barTempl.h: 4003

**Related method**

**Since**

[SDK LATEEST VERSION > 2.0](#)

### FRRibbonSliderGetMouseStatus

**Syntax**

```
FS_INT32 FRRibbonSliderGetMouseStatus (
    FR\_RibbonSlider ribbonSlider
);
```

**Description**

Gets the mouse status. -1 - none, 0 - LButtonDown, 1 - LButtonUp, 2 - LButtonDown & MouseMove.

**Parameter**

---

ribbonSlider	[In] The input ribbon slider object.
--------------	--------------------------------------

---

**Return**

The mouse status. -1 - none, 0 - LButtonDown, 1 - LButtonUp, 2 - LButtonDown & MouseMove.

**Head file reference**

fr\_barTempl.h: 4110

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonSliderGetPos****Syntax**

```
FS_INT32 FRRibbonSliderGetPos (
    FR\_RibbonSlider ribbonSlider
);
```

**Description**

Gets the position of the slider.

**Parameter**


---

ribbonSlider	[In] The input ribbon slider object.
--------------	--------------------------------------

---

**Return**

The position of the slider.

**Head file reference**

fr\_barTempl.h: 4075

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonSliderGetRangeMax****Syntax**

```
FS_INT32 FRRibbonSliderGetRangeMax (
    FR\_RibbonSlider ribbonSlider
);
```

**Description**

Gets the maximum value of the range.

#### Parameter

---

ribbonSlider	[In] The input ribbon slider object.
--------------	--------------------------------------

---

#### Return

The maximum value of the range.

#### Head file reference

fr\_barTempl.h: 4051

#### Related method

##### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonSliderGetRangeMin

#### Syntax

```
FS_INT32 FRRibbonSliderGetRangeMin (
    FR\_RibbonSlider ribbonSlider
);
```

#### Description

Gets the minimum value of the range.

#### Parameter

---

ribbonSlider	[In] The input ribbon slider object.
--------------	--------------------------------------

---

#### Return

The minimum value of the range.

#### Head file reference

fr\_barTempl.h: 4040

#### Related method

##### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonSliderSetPos

#### Syntax

```
void FRRibbonSliderSetPos (
    FR\_RibbonSlider ribbonSlider,
    FS\_INT32 nPos,
```

```
FS_BOOL bRedraw  
);
```

**Description**

Sets the position of the slider.

**Parameter**

ribbonSlider	[In] The input ribbon slider object.
nPos	[In] The input position.
bRedraw	[In] Sets TRUE to redraw the slider, otherwise not.

**Return**

void

**Head file reference**

fr\_barTempl.h: 4062

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonSliderSetRange****Syntax**

```
void FRRibbonSliderSetRange (  
    FR_RibbonSlider ribbonSlider,  
    FS_INT32 nMin,  
    FS_INT32 nMax  
)
```

**Description**

Sets the range of the slider.

**Parameter**

ribbonSlider	[In] The input ribbon slider object.
nMin	[In] The minimum value of the range.
nMax	[In] The maximum value of the range.

**Return**

void

**Head file reference**

fr\_barTempl.h: 4027

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonSliderSetStyle****Syntax**

```
void FRRibbonSliderSetStyle (
    FR\_RibbonSlider ribbonSlider,
    FS\_DWORD dwStyle
);
```

**Description**

Sets the style of the ribbon slider.

**Parameter**

---

ribbonSlider	[In] The input ribbon slider object.
--------------	--------------------------------------

---

dwStyle	[In] The input style of the ribbon slider. References to <i>MFC CSliderCtrl</i> .
---------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4098

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonSliderSetWidth****Syntax**

```
void FRRibbonSliderSetWidth (
    FR\_RibbonSlider ribbonSlider,
    FS\_INT32 nWidth
);
```

**Description**

Sets the width of the ribbon slider.

**Parameter**

---

ribbonSlider	[In] The input ribbon slider object.
--------------	--------------------------------------

---

nWidth	[In] The input width of the ribbon slider.
--------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4086

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonSliderSetZoomButtons****Syntax**

```
void FRRibbonSliderSetZoomButtons (
    FR\_RibbonSlider ribbonSlider,
    FS\_BOOL bSet
);
```

**Description**

Sets whether to add the zoom buttons or not.

**Parameter**

---

ribbonSlider	[In] The input ribbon slider object.
--------------	--------------------------------------

---

bSet	[In] Sets TRUE to add the zoom buttons.
------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4015

**Related method**

Since  
[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FR\_RibbonStyleButton

### [Used by](#)

#### Description

The ribbon style button. You can modify the MFC button to the ribbon style button.

#### Used by

[FRRibbonStyleButtonGetMFCButton](#)  
[FRRibbonStyleButtonRelease](#)  
[FRRibbonStyleButtonSetAlignStyle](#)  
[FRRibbonStyleButtonSetFlatStyle](#)  
[FRRibbonStyleButtonSetImage](#)  
[FRRibbonStyleButtonSetImage2](#)  
[FRRibbonStyleButtonSetImageSide](#)  
[FRRibbonStyleButtonSetVertMargin](#)

#### Functions

##### Functions summary

[FRRibbonStyleButtonGetMFCButton](#)  
Gets the pointer to the MFC class.  
[FRRibbonStyleButtonRelease](#)  
Releases the ribbon style button.  
[FRRibbonStyleButtonSetAlignStyle](#)  
Sets the alignment type of the ribbon style button.  
[FRRibbonStyleButtonSetFlatStyle](#)  
Sets the flat style of the ribbon style button.  
[FRRibbonStyleButtonSetImage](#)  
Sets the icon of the ribbon style button.  
[FRRibbonStyleButtonSetImage2](#)  
Sets the icon of the ribbon style button.  
[FRRibbonStyleButtonSetImageSide](#)  
Sets the icon side of the ribbon style button.  
[FRRibbonStyleButtonSetVertMargin](#)  
Sets the margin in the vertical direction.

##### Functions detail

###### [FRRibbonStyleButtonGetMFCButton](#)

###### Syntax

```
void* FRRibbonStyleButtonGetMFCButton (
```

[FR\\_RibbonStyleButton](#) button

);

**Description**

Gets the pointer to the MFC class.

**Parameter**

---

button	[In] The input ribbon style button object.
--------	--

---

**Return**

The pointer to the MFC class.

**Head file reference**

fr\_barTempl.h: 4670

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonStyleButtonRelease****Syntax**

```
void FRRibbonStyleButtonRelease (
    FR\_RibbonStyleButton button
);
```

**Description**

Releases the ribbon style button.

**Parameter**

---

button	[In] The input ribbon style button object.
--------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4681

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonStyleButtonSetAlignStyle**

**Syntax**

```
void FRRibbonStyleButtonSetAlignStyle (
    FR\_RibbonStyleButton button,
    FRRibbonStyleTextAlignType nAlignType
);
```

**Description**

Sets the alignment type of the ribbon style button.

**Parameter**

button	[In] The input ribbon style button object.
nAlignType	[In] The input alignment type of the ribbon style button.

**Return**

void

**Head file reference**

fr\_barTempl.h: 4622

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonStyleButtonSetFlatStyle****Syntax**

```
void FRRibbonStyleButtonSetFlatStyle (
    FR\_RibbonStyleButton button,
    FRRibbonStyleFlatType nFlatStyle
);
```

**Description**

Sets the flat style of the ribbon style button.

**Parameter**

button	[In] The input ribbon style button object.
nFlatStyle	[In] The input flat style of the ribbon style button.

**Return**

void

**Head file reference**

fr\_barTempl.h: 4646

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonStyleButtonSetImage****Syntax**

```
void FRRibbonStyleButtonSetImage (
    FR\_RibbonStyleButton button,
    FS\_DIBitmap pBitmap
);
```

**Description**

Sets the icon of the ribbon style button.

**Parameter**

---

button	[In] The input ribbon style button object.
--------	--

---

pBitmap	[In] The input icon of the ribbon style button.
---------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4610

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonStyleButtonSetImage2****Syntax**

```
void FRRibbonStyleButtonSetImage2 (
    FR\_RibbonStyleButton button,
    FS\_DIBitmap pBitmap,
    FS\_INT32 cx,
    FS\_INT32 cy
);
```

**Description**

Sets the icon of the ribbon style button.

**Parameter**

button	[In] The input ribbon style button object.
pBitmap	[In] The input icon of the ribbon style button.
cx	[In] The width of the bitmap when the DPI is 100%. The default value is 32.
cy	[In] The height of the bitmap when the DPI is 100%. The default value is 32.

**Return**

void

**Head file reference**

fr\_barTempl.h: 4692

**Related method****Since**[SDK LATEEST VERSION > 3.0](#)**FRRibbonStyleButtonSetImageSide****Syntax**

```
void FRRibbonStyleButtonSetImageSide (  
    FR\_RibbonStyleButton button,  
    FRRibbonStyleImageSide nImageSide  
) ;
```

**Description**

Sets the icon side of the ribbon style button.

**Parameter**

button	[In] The input ribbon style button object.
nImageSide	[In] The input icon side of the ribbon style button.

**Return**

void

**Head file reference**

fr\_barTempl.h: 4634

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

## FRRibbonStyleButtonSetVertMargin

#### Syntax

```
void FRRibbonStyleButtonSetVertMargin (
    FR\_RibbonStyleButton button,
    FS\_INT32 nVerMargin
);
```

#### Description

Sets the margin in the vertical direction.

#### Parameter

button	[In] The input ribbon style button object.
nVerMargin	[In] The input margin in the vertical direction.

#### Return

void

#### Head file reference

fr\_barTempl.h: 4658

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

## FR\_RibbonStyleListBox

### [Used by](#)

#### Description

The ribbon style list box. You can modify the MFC list box to the ribbon style list box.

#### Used by

[FRRibbonStyleListBoxAddCaption](#)

[FRRibbonStyleListBoxAddImage](#)

---

[FRRibbonStyleListBoxAddImage2](#)  
[FRRibbonStyleListBoxAddString](#)  
[FRRibbonStyleListBoxClear](#)  
[FRRibbonStyleListBoxDeleteString](#)  
[FRRibbonStyleListBoxEnablePins](#)  
[FRRibbonStyleListBoxGetItemData](#)  
[FRRibbonStyleListBoxGetItemPinnedState](#)  
[FRRibbonStyleListBoxGetMFCLListBox](#)  
[FRRibbonStyleListBoxHasPins](#)  
[FRRibbonStyleListBoxInsertString](#)  
[FRRibbonStyleListBoxIsSeparatorItem](#)  
[FRRibbonStyleListBoxRedrawList](#)  
[FRRibbonStyleListBoxRelease](#)  
[FRRibbonStyleListBoxSetClickItemProc](#)  
[FRRibbonStyleListBoxSetClickPinProc](#)  
[FRRibbonStyleListBoxSetClientData](#)  
[FRRibbonStyleListBoxSetItemData](#)  
[FRRibbonStyleListBoxSetItemPinned](#)

## Callbacks

### Callbacks summary

#### [FRClickItemProc](#)

A callback for Ribbon style list box.

#### [FRClickPinProc](#)

A callback for Ribbon style list box.

### Callbacks detail

#### [FRClickItemProc](#)

##### Syntax

```
typedef void (*FRClickItemProc)(  
    FS_INT32 nIndex,  
    void clientData  
)
```

##### Description

A callback for Ribbon style list box. It is called when the item of the Ribbon style list box is clicked.

##### Parameter

---

nIndex	[In] The item index of the Ribbon style list.
--------	---

---

clientData	[In] User-supplied data.
------------	--------------------------

---

##### Return

void

**Head file reference**

fr\_barExpT.h: 716

**Related method**[FRRibbonStyleListBoxSetClickItemProc](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.2.2](#)**FRClickPinProc****Syntax**

```
typedef void (*FRClickPinProc)(  
    FS_INT32 nIndex,  
    void clientData  
)
```

**Description**

A callback for Ribbon style list box. It is called when the item pin button of the Ribbon style list box is clicked.

**Parameter**

---

nIndex	[In] The item index of the Ribbon style list.
--------	---

---

clientData	[In] User-supplied data.
------------	--------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 732

**Related method**[FRRibbonStyleListBoxSetClickPinProc](#)**Since**[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

## Functions

### Functions summary

**[FRRibbonStyleListBoxAddCaption](#)**

Adds the caption of the ribbon style list box.

**[FRRibbonStyleListBoxAddImage](#)**

Adds a 16\*16 icon list to the ribbon style list box.

**FRRibbonStyleListBoxAddImage2**

Adds a 16\*16 icon list to the ribbon style list box.

**FRRibbonStyleListBoxAddString**

Adds a new string item to the ribbon style list box.

**FRRibbonStyleListBoxClear**

Clears the ribbon style list box.

**FRRibbonStyleListBoxDeleteString**

Deletes the string item by index.

**FRRibbonStyleListBoxEnablePins**

Whether to enable the pin buttons or not.

**FRRibbonStyleListBoxGetItemData**

Gets the item data of the ribbon style list box.

**FRRibbonStyleListBoxGetItemPinnedState**

Gets the status of the pin of the specified item.

**FRRibbonStyleListBoxGetMFCLListBox**

Gets the pointer to the MFC class.

**FRRibbonStyleListBoxHasPins**

Whether the ribbon style list box has pins or not.

**FRRibbonStyleListBoxInsertString**

Inserts a string into the ribbon style list box.

**FRRibbonStyleListBoxIsSeparatorItem**

Checks whether the item is a separator or not.

**FRRibbonStyleListBoxRedrawList**

Redraws the ribbon style list box.

**FRRibbonStyleListBoxRelease**

Releases the ribbon style list box.

**FRRibbonStyleListBoxSetClickItemProc**

Sets the callback which is called when the item of the Ribbon style list box is clicked.

**FRRibbonStyleListBoxSetClickPinProc**

Sets the callback which is called when the item pin button of the Ribbon style list box is clicked.

**FRRibbonStyleListBoxSetClientData**

Sets the client data.

**FRRibbonStyleListBoxSetItemData**

Sets a 32-bit value associated with the specified item in a ribbon style list box.

**FRRibbonStyleListBoxSetItemPinned**

Sets the status of the pin of the specified item.

## Functions detail

### **FRRibbonStyleListBoxAddCaption**

#### **Syntax**

```
void FRRibbonStyleListBoxAddCaption (
    FR_RibbonStyleListBox listBox,
    FS_LPCWSTR lpwsCaption
);
```

#### **Description**

Adds the caption of the ribbon style list box.

**Parameter**


---

listBox	[In] The input ribbon style list box object.
---------	--

---

lpwsCaption	[In] The input caption of the ribbon style list box.
-------------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4740

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonStyleListBoxAddImage****Syntax**

```
FS_INT32 FRRibbonStyleListBoxAddImage (
    FR\_RibbonStyleListBox listBox,
    FS\_DIBitmap bitmap
);
```

**Description**

Adds a 16\*16 icon list to the ribbon style list box.

**Parameter**


---

listBox	[In] The input ribbon style list box object.
---------	--

---

bitmap	[In] The input 16*16 icon list of the ribbon style list box.
--------	--

---

**Return**

The count of the icons.

**Head file reference**

fr\_barTempl.h: 4714

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonStyleListBoxAddImage2**

**Syntax**

```
FS_INT32 FRRibbonStyleListBoxAddImage2 (
    FR.RibbonStyleListBox listBox,
    FS\_DIBitmap bitmap,
    FS\_INT32 cx,
    FS\_INT32 cy
);
```

**Description**

Adds a 16\*16 icon list to the ribbon style list box.

**Parameter**


---

listBox	[In] The input ribbon style list box object.
---------	--

---

bitmap	[In] The input 16*16 icon list of the ribbon style list box.
--------	--

---

cx	[In] The width of the bitmap when the DPI is 100%. The default value is 32.
----	---

---

cy	[In] The height of the bitmap when the DPI is 100%. The default value is 32.
----	--

---

**Return**

The count of the icons.

**Head file reference**

fr\_barTempl.h: 4826

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 3.0](#)

**FRRibbonStyleListBoxAddString****Syntax**

```
FS_INT32 FRRibbonStyleListBoxAddString (
    FR.RibbonStyleListBox listBox,
    FS\_LPCWSTR lpwsItem,
    FS\_INT32 nImageIndex
);
```

**Description**

Adds a new string item to the ribbon style list box.

**Parameter**

---

listBox	[In] The input ribbon style list box object.
lpwsItem	[In] The input string separated by '\n'. For example, "Foxit Reader\nThe Best PDF Reader\nOne of the most popular PDF reader", the string item will show two lines and the tooltip will be "One of the most popular PDF reader".
nImageIndex	[In] The specified index of the icon list.

---

**Return**

The index of the string item.

**Head file reference**

fr\_barTempl.h: 4726

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonStyleListBoxClear****Syntax**

```
void FRRibbonStyleListBoxClear (
    FR\_RibbonStyleListBox listBox
);
```

**Description**

Clears the ribbon style list box.

**Parameter**


---

listBox	[In] The input ribbon style list box object.
---------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 4949

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRRibbonStyleListBoxDeleteString****Syntax**

```
FS_INT32 FRRibbonStyleListBoxDeleteString (
    FR.RibbonStyleListBox listBox,
    unsigned int nIndex
);
```

**Description**

Deletes the string item by index.

**Parameter**

---

listBox	[In] The input ribbon style list box object.
---------	--

---

nIndex	[In] The specified index.
--------	---------------------------

---

**Return**

The index of the currently selected item.

**Head file reference**

fr\_barTempl.h: 4752

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonStyleListBoxEnablePins****Syntax**

```
void FRRibbonStyleListBoxEnablePins (
    FR.RibbonStyleListBox listBox,
    FS\_BOOL bEnable
);
```

**Description**

Whether to enable the pin buttons or not.

**Parameter**

---

listBox	[In] The input ribbon style list box object.
---------	--

---

bEnable	[In] TRUE means enable the pin buttons, otherwise not.
---------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 4840

#### Related method

##### Since

[SDK LATEEST VERSION > 7.2.2](#)

### FRRibbonStyleListBoxGetItemData

#### Syntax

```
FS_DWORD FRRibbonStyleListBoxGetItemData (
    FR\_RibbonStyleListBox listBox,
    FS\_INT32 nIndex
);
```

#### Description

Gets the item data of the ribbon style list box.

#### Parameter

listBox	[In] The input ribbon style list box object.
nIndex	[In] The specified index.

#### Return

The item data of the ribbon style list box.

#### Head file reference

fr\_barTempl.h: 4779

#### Related method

##### Since

[SDK LATEEST VERSION > 2.0](#)

### FRRibbonStyleListBoxGetItemPinnedState

#### Syntax

```
FS_INT32 FRRibbonStyleListBoxGetItemPinnedState (
    FR\_RibbonStyleListBox listBox,
    FS\_INT32 nIndex
);
```

#### Description

Gets the status of the pin of the specified item.

#### Parameter

---

listBox	[In] The input ribbon style list box object.
---------	--

---

nIndex	[In] The specified index of the item.
--------	---------------------------------------

---

**Return**

The status of the pin, 0 for not pinned, 1 for pinned, 2 for not display the pin button.

**Head file reference**

fr\_barTempl.h: 4877

**Related method****Since**

[SDK LATEEST VERSION > 7.2.2](#)

**FRRibbonStyleListBoxGetMFCLListBox****Syntax**

```
void* FRRibbonStyleListBoxGetMFCLListBox (
    FR\_RibbonStyleListBox listBox
);
```

**Description**

Gets the pointer to the MFC class.

**Parameter**


---

listBox	[In] The input ribbon style list box object.
---------	--

---

**Return**

The pointer to the MFC class.

**Head file reference**

fr\_barTempl.h: 4804

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonStyleListBoxHasPins****Syntax**

```
FS_BOOL FRRibbonStyleListBoxHasPins (
    FR\_RibbonStyleListBox listBox
);
```

**Description**

Whether the ribbon style list box has pins or not.

**Parameter**

---

listBox	[In] The input ribbon style list box object.
---------	--

---

**Return**

TRUE means the ribbon style list box has pins, otherwise not.

**Head file reference**

fr\_barTempl.h: 4852

**Related method****Since**

[SDK LATEEST VERSION > 7.2.2](#)

**FRRibbonStyleListBoxInsertString****Syntax**

```
FS_INT32 FRRibbonStyleListBoxInsertString (
    FR\_RibbonStyleListBox listBox,
    FS_INT32 nIndex,
    FS_LPCWSTR lpwsItem,
    FS_INT32 nImageIndex
);
```

**Description**

Inserts a string into the ribbon style list box.

**Parameter**

---

listBox	[In] The input ribbon style list box object.
---------	--

---

nIndex	[In] Specifies the zero-based index of the position to insert the string. If this parameter is $\text{ijklC1}$ , the string is added to the end of the list.
--------	--

---

lpwsItem	[In] The input string separated by '\n'. For example, "Foxit Reader\nThe Best PDF Reader\nOne of the most popular PDF reader", the string item will show two lines and the tooltip will be "One of the most popular PDF reader".
----------	--

---

nImageIndex	[In] The specified index of the icon list.
-------------	--

---

**Return**

The zero-based index of the position at which the string was inserted.

**Head file reference**

fr\_barTempl.h: 4764

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRRibbonStyleListBoxIsSeparatorItem****Syntax**

```
FS_BOOL FRRibbonStyleListBoxIsSeparatorItem (
    FR\_RibbonStyleListBox listBox,
    FS_INT32 nIndex
);
```

**Description**

Checks whether the item is a separator or not.

**Parameter**

---

listBox	[In] The input ribbon style list box object.
---------	--

---

nIndex	[In] The specified index of the item.
--------	---------------------------------------

---

**Return**

TRUE if the item is a separator, otherwise not.

**Head file reference**

fr\_barTempl.h: 4889

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FRRibbonStyleListBoxRedrawList****Syntax**

```
void FRRibbonStyleListBoxRedrawList (
    FR\_RibbonStyleListBox listBox
);
```

**Description**

Redraws the ribbon style list box.

#### Parameter

---

listBox	[In] The input ribbon style list box object.
---------	--

---

#### Return

void.

#### Head file reference

fr\_barTempl.h: 4901

#### Related method

##### Since

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

## FRRibbonStyleListBoxRelease

#### Syntax

```
void FRRibbonStyleListBoxRelease (
    FR\_RibbonStyleListBox listBox
);
```

#### Description

Releases the ribbon style list box.

#### Parameter

---

listBox	[In] The input ribbon style list box object.
---------	--

---

#### Return

void

#### Head file reference

fr\_barTempl.h: 4815

#### Related method

##### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRRibbonStyleListBoxSetClickItemProc

#### Syntax

```
void FRRibbonStyleListBoxSetClickItemProc (
    FR\_RibbonStyleListBox listBox,
    FRClickItemProc proc
```

);

**Description**

Sets the callback which is called when the item of the Ribbon style list box is clicked.

**Parameter**

---

listBox	[In] The input ribbon style list box object.
---------	--

---

proc	[In] A callback for Ribbon style list box.
------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 4912

**Related method****Since**

[SDK LATEEST VERSION > 7.2.2](#)

**FRRibbonStyleListBoxSetClickPinProc**

**Syntax**

```
void FRRibbonStyleListBoxSetClickPinProc (
    FR\_RibbonStyleListBox listBox,
    FRClickPinProc proc
);
```

**Description**

Sets the callback which is called when the item pin button of the Ribbon style list box is clicked.

**Parameter**

---

listBox	[In] The input ribbon style list box object.
---------	--

---

proc	[In] A callback for Ribbon style list box.
------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 4924

**Related method****Since**[SDK LATEEST VERSION > 7.2.2](#)**FRRibbonStyleListBoxSetClientData****Syntax**

```
void FRRibbonStyleListBoxSetClientData (
    FR\_RibbonStyleListBox listBox,
    void* pData,
    FRFreeDataProc callback
);
```

**Description**

Sets the client data.

**Parameter**

---

listBox	[In] The input ribbon style list box object.
---------	--

---

pData	[In] The input client data.
-------	-----------------------------

---

callback	[In] The callback will be invoked to free the client data.
----------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 4936

**Related method****Since**[SDK LATEEST VERSION > 7.2.2](#)**FRRibbonStyleListBoxSetItemData****Syntax**

```
FS_INT32 FRRibbonStyleListBoxSetItemData (
    FR\_RibbonStyleListBox listBox,
    FS\_INT32 nIndex,
    FS\_DWORD dwItemData
);
```

**Description**

Sets a 32-bit value associated with the specified item in a ribbon style list box.

**Parameter**


---

listBox	[In] The input ribbon style list box object.
nIndex	[In] The specified index.
dwItemData	[In] The input item data.

---

**Return**

LB\_ERR if an error occurs.

**Head file reference**

fr\_barTempl.h: 4791

**Related method****Since**

[SDK LATEEST VERSION > 2.0](#)

**FRRibbonStyleListBoxSetItemPinned****Syntax**

```
void FRRibbonStyleListBoxSetItemPinned (
    FR\_RibbonStyleListBox listBox,
    FS\_INT32 nIndex,
    FS\_INT32 nPin,
    FS\_BOOL bRedraw
);
```

**Description**

Sets the status of the pin of the specified item.

**Parameter**


---

listBox	[In] The input ribbon style list box object.
nIndex	[In] The specified index of the item.
nPin	[In] The status of the pin, 0 for not pinned, 1 for pinned, 2 for not display the pin button.
bRedraw	[In] Whether to redraw the list box or not.

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 4863

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

## FR\_RibbonStyleStatic

### [Used by](#)

### Description

The ribbon style static box. You can modify the MFC static box to the ribbon style static box.

### Used by

[FRRibbonStyleStaticGetMFCStatic](#)  
[FRRibbonStyleStaticRelease](#)  
[FRRibbonStyleStaticSetBackgroundColor](#)  
[FRRibbonStyleStaticSetFontStyle](#)  
[FRRibbonStyleStaticSetImage](#)  
[FRRibbonStyleStaticSetImage2](#)  
[FRRibbonStyleStaticSetTextColor](#)  
[FRRibbonStyleStaticSetTextDrawFormat](#)

### Functions

#### Functions summary

[FRRibbonStyleStaticGetMFCStatic](#)

Gets the pointer to the corresponding MFC class.

[FRRibbonStyleStaticRelease](#)

Releases the ribbon style static box.

[FRRibbonStyleStaticSetBackgroundColor](#)

Sets the background color of the ribbon style static box.

[FRRibbonStyleStaticSetFontStyle](#)

Sets the font and the style of the ribbon style static box.

[FRRibbonStyleStaticSetImage](#)

Sets the icon of the ribbon style static box.

[FRRibbonStyleStaticSetImage2](#)

Sets the icon of the ribbon style static box.

[FRRibbonStyleStaticSetTextColor](#)

Sets the color of the ribbon style static box.

[FRRibbonStyleStaticSetTextDrawFormat](#)

Sets the text format of the ribbon style static box.

## Functions detail

### FRRibbonStyleStaticGetMFCStatic

#### Syntax

```
void* FRRibbonStyleStaticGetMFCStatic (
    FR\_RibbonStyleStatic stcObj
);
```

#### Description

Gets the pointer to the corresponding MFC class.

#### Parameter

---

stcObj	[In] The input ribbon style static box object.
--------	--

---

#### Return

The pointer to the corresponding MFC class.

#### Head file reference

fr\_barTempl.h: 4980

#### Related method

#### Since

[SDK LATEEST VERSION > 2.0](#)

### FRRibbonStyleStaticRelease

#### Syntax

```
void FRRibbonStyleStaticRelease (
    FR\_RibbonStyleStatic stcObj
);
```

#### Description

Releases the ribbon style static box.

#### Parameter

---

stcObj	[In] The input ribbon style static box object.
--------	--

---

#### Return

void

#### Head file reference

fr\_barTempl.h: 5043

#### Related method

**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonStyleStaticSetBackgroundColor****Syntax**

```
void FRRibbonStyleStaticSetBackgroundColor (
    FR\_RibbonStyleStatic stcObj,
    FS\_DWORD dwBgColor
);
```

**Description**

Sets the background color of the ribbon style static box.

**Parameter**

---

stcObj	[In] The input ribbon style static box object.
--------	--

---

dwBgColor	[In] The input background color value.
-----------	--

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 5019

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonStyleStaticSetFontStyle****Syntax**

```
void FRRibbonStyleStaticSetFontStyle (
    FR\_RibbonStyleStatic stcObj,
    FS\_INT32 nFontSize,
    FS\_BOOL bBold,
    FS\_BOOL bItalic,
    FS\_BOOL bUnderlined,
    FS\_BOOL bStrikethrough
);
```

**Description**

Sets the font and the style of the ribbon style static box.

**Parameter**

---

stcObj	[In] The input ribbon style static box object.
nFontSize	[In] The input font size of the ribbon style static box.
bBold	[In] TRUE if the ribbon style static box uses the bold, otherwise not.
bItalic	[In] TRUE if the ribbon style static box uses the italic, otherwise not.
bUnderlined	[In] TRUE if the ribbon style static box is underlined, otherwise not.
bStrikethrough	[In] TRUE if the ribbon style static box is stricken through, otherwise not.

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 4991

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRRibbonStyleStaticSetImage****Syntax**

```
void FRRibbonStyleStaticSetImage (
    FR\_RibbonStyleStatic stcObj,
    FS\_DIBitmap pBitmap
);
```

**Description**

Sets the icon of the ribbon style static box.

**Parameter**


---

stcObj	[In] The input ribbon style static box object.
pBitmap	[In] The input icon of the ribbon style static box.

---

**Return**

---

```
void
```

**Head file reference**

fr\_barTempl.h: 4968

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRRibbonStyleStaticSetImage2****Syntax**

```
void FRRibbonStyleStaticSetImage2 (
    FR\_RibbonStyleStatic stcObj,
    FS\_DIBitmap pBitmap,
    FS\_INT32 cx,
    FS\_INT32 cy
);
```

**Description**

Sets the icon of the ribbon style static box.

**Parameter**

stcObj	[In] The input ribbon style static box object.
pBitmap	[In] The input icon of the ribbon style static box.
cx	[In] The width of the bitmap when the DPI is 100%. The default value is 32.
cy	[In] The height of the bitmap when the DPI is 100%. The default value is 32.

**Return**

void

**Head file reference**

fr\_barTempl.h: 5054

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 3.0](#)

## FRRibbonStyleStaticSetTextColor

### Syntax

```
void FRRibbonStyleStaticSetTextColor (
    FR\_RibbonStyleStatic stcObj,
    FS\_DWORD dwTextColor
);
```

### Description

Sets the color of the ribbon style static box.

### Parameter

---

stcObj	[In] The input ribbon style static box object.
--------	--

---

dwTextColor	[In] The input color value.
-------------	-----------------------------

---

### Return

void

### Head file reference

fr\_barTempl.h: 5007

### Related method

#### Since

[SDK LATEEST VERSION > 2.0](#)

## FRRibbonStyleStaticSetTextDrawFormat

### Syntax

```
void FRRibbonStyleStaticSetTextDrawFormat (
    FR\_RibbonStyleStatic stcObj,
    unsigned int nFormat
);
```

### Description

Sets the text format of the ribbon style static box.

### Parameter

---

stcObj	[In] The input ribbon style static box object.
--------	--

---

nFormat	[In] The input format. References to MFC description such as DT_BOTTOM.
---------	---

---

### Return



void

**Head file reference**

fr\_barTempl.h: 5031

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FR\_ScrollBarThumbnailView

### [Used by](#)

#### Description

##### Used by

[FRScrollBarThumbnailViewGetCurPageIndex](#)  
[FRScrollBarThumbnailViewGetPageRect](#)  
[FRScrollBarThumbnailViewGetPDPPage](#)

#### Functions

##### Functions summary

[FRScrollBarThumbnailViewGetCurPageIndex](#)

Gets the index of page displayed in the scroll bar thumbnail view.

[FRScrollBarThumbnailViewGetPageRect](#)

Gets the rectangle of specified visible page.

[FRScrollBarThumbnailViewGetPDPPage](#)

Gets a [FPD\\_Page](#) object.

##### Functions detail

###### [FRScrollBarThumbnailViewGetCurPageIndex](#)

###### Syntax

```
FS_INT32 FRScrollBarThumbnailViewGetCurPageIndex (
    FR\_ScrollBarThumbnailView tv
);
```

###### Description

Gets the index of page displayed in the scroll bar thumbnail view.

###### Parameter

---

tv

[In] The input scrollbar thumbnail view.

---

**Return**

The index of page displayed in the scroll bar thumbnail view.

**Head file reference**

fr\_viewTempl.h: 1281

**Related method****Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FRScrollBarThumbnailViewGetPageRect****Syntax**

```
FS_Rect FRScrollBarThumbnailViewGetPageRect (
    FR\_ScrollBarThumbnailView tv
);
```

**Description**

Gets the rectangle of specified visible page.

**Parameter**

---

tv	[In] The input scrollbar thumbnail view.
----	--

---

**Return**

The rectangle of specified visible page.

**Head file reference**

fr\_viewTempl.h: 1292

**Related method****Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FRScrollBarThumbnailViewGetPDPage****Syntax**

```
FPD_Page FRScrollBarThumbnailViewGetPDPage (
    FR\_ScrollBarThumbnailView tv
);
```

**Description**

Gets a [FPD\\_Page](#) object.

**Parameter**

---

tv	[In] The input scrollbar thumbnail view.
----	--

---

**Return**

The [FPD\\_Page](#) object.

**Head file reference**

fr\_viewTempl.h: 1303

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

## FR\_SpellCheck

### [Return from Used by](#)

#### Description

The [FR\\_SpellCheck](#) object can be used to check the spelling. See [FRSpellCheckNew](#).

#### Returned from

### [FRSpellCheckNew](#)

#### Used by

[FRSpellCheckAddDic](#)  
[FRSpellCheckCheckString](#)  
[FRSpellCheckCheckWord](#)  
[FRSpellCheckDestroy](#)  
[FRSpellCheckIgnore](#)  
[FRSpellCheckSuggestWords](#)

#### Functions

##### Functions summary

###### [FRSpellCheckAddDic](#)

Adds the new words to the dictionary.

###### [FRSpellCheckCheckString](#)

Checks the spelling of the input byte string and receives the errors.

###### [FRSpellCheckCheckWord](#)

Checks whether the input words are correct or not.

###### [FRSpellCheckDestroy](#)

Destroys the input [FR\\_SpellCheck](#) object.

###### [FRSpellCheckIgnore](#)

Ignores the input words that will not be checked.

###### [FRSpellCheckNew](#)

Creates a [FR\\_SpellCheck](#) object used to check spelling.

#### [FRSpellCheckSuggestWords](#)

Input the words to get its suggested words.

### Functions detail

#### [FRSpellCheckAddDic](#)

##### **Syntax**

```
void FRSpellCheckAddDic (
    FR\_SpellCheck spellCheck,
    FS\_LPCWSTR sWord
);
```

##### **Description**

Adds the new words to the dictionary.

##### **Parameter**

---

spellCheck	[In] The input <a href="#">FR_SpellCheck</a> object.
------------	--

---

sWord	[In] The input new words.
-------	---------------------------

---

##### **Return**

void

##### **Head file reference**

fr\_appTempl.h: 3604

##### **Related method**

#### [FRSpellCheckNew](#)

##### **Since**

[SDK LATEEST VERSION > 2.0](#)

#### [FRSpellCheckCheckString](#)

##### **Syntax**

```
void FRSpellCheckCheckString (
    FR\_SpellCheck spellCheck,
    csString,
    FS\_WideStringArray* outErrBufArr
);
```

##### **Description**

Checks the spelling of the input byte string and receives the errors.

##### **Parameter**

---

spellCheck	[In] The input <a href="#">FR_SpellCheck</a> object.
------------	--

---

csString	[In] The input byte string that will be checked.
----------	--

---

outErrBufArr	[Out] It receives the errors.
--------------	-------------------------------

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 3591

**Related method**[FRSpellCheckCheckWord](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRSpellCheckCheckWord****Syntax**

```
FS_BOOL FRSpellCheckCheckWord (
    FR\_SpellCheck spellCheck,
    FS\_LPCWSTR sWord
);
```

**Description**

Checks whether the input words are correct or not.

**Parameter**


---

spellCheck	[In] The input <a href="#">FR_SpellCheck</a> object.
------------	--

---

sWord	[In] The input words to be checked.
-------	-------------------------------------

---

**Return**[TRUE](#) if the words are checked correctly, otherwise not.**Head file reference**

fr\_appTempl.h: 3538

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRSpellCheckDestroy

### Syntax

```
void FRSpellCheckDestroy (
    FR\_SpellCheck spellCheck
);
```

### Description

Destroys the input [FR\\_SpellCheck](#) object.

### Parameter

---

spellCheck	[In] The input <a href="#">FR_SpellCheck</a> object.
------------	--

---

### Return

void

### Head file reference

fr\_appTempl.h: 3537

### Related method

### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRSpellCheckIgnore

### Syntax

```
void FRSpellCheckIgnore (
    FR\_SpellCheck spellCheck,
    FS\_LPCWSTR sWord
);
```

### Description

Ignores the input words that will not be checked.

### Parameter

---

spellCheck	[In] The input <a href="#">FR_SpellCheck</a> object.
------------	--

---

sWord	[In] The input words to be ignored.
-------	-------------------------------------

---

### Return

void

### Head file reference

fr\_appTempl.h: 3579

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRSpellCheckNew****Syntax**

```
FR_SpellCheck FRSpellCheckNew (
    FS\_LPCWSTR lpwsModuleName
);
```

**Description**

Creates a [FR\\_SpellCheck](#) object used to check spelling.

**Parameter**

---

lpwsModuleName	[In] The input module name.
----------------	-----------------------------

---

**Return**

A new [FR\\_SpellCheck](#) object.

**Head file reference**

fr\_appTempl.h: 3531

**Related method**[FRSpellCheckDestroy](#)[FRSpellCheckCheckWord](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRSpellCheckSuggestWords****Syntax**

```
void FRSpellCheckSuggestWords (
    FR\_SpellCheck spellCheck,
    FS\_LPCWSTR sWord,
    FS\_WideStringArray* outSuggest
);
```

**Description**

Input the words to get its suggested words.

**Parameter**

---

spellCheck	[In] The input <a href="#">FR_SpellCheck</a> object.
sWord	[In] The input words.
outSuggest	[Out] It receives the suggested words.

---

**Return**  
void

**Head file reference**  
fr\_appTempl.h: 3566

#### Related method

**Since**  
[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FR\_StatusBar

### [Used by](#)

#### Description

#### Used by

### [FRStatusBarAddWndEx](#)

#### Functions

#### Functions summary

### [FRStatusBarAddWndEx](#)

Adds a windows to the status bar.

### [FRStatusBarReset](#)

Resets the status bar.

### [FRStatusBarSetZoomSliderRange](#)

Sets the zoom slider range.

### [FRStatusBarSetZoomSliderPos](#)

Sets the position of the zoom slider.

### [FRStatusBarGetZoomSliderPos](#)

Gets the position of the zoom slider.

### [FRStatusBarSetZoomPaneText](#)

Sets the text of the zoom pane.

### [FRStatusBarSetComboBoxPageCount](#)

Sets the page count of the page combo box.

### [FRStatusBarSelectComboBoxItem](#)

Selects the specified page combo box item.

**FRStatusBarGetComboBoxPageIndex**

Gets the page index of the page combo box.

**FRStatusBarGetComboBoxPageText**

Gets the text of the page combo box.

**FRStatusBarGetBkGroundColor**

Gets the back ground color of status bar in ribbon mode.

**FRStatusBarGetBkGroundPath**

Gets the path of the back ground color picture of the status bar in classic mode.

**FRStatusBarRecalcLayout**

Recalculates the layout of the status bar.

**FRStatusBarSetComboBoxPageCount2**

Sets the page count of the page combo box.

**FRStatusBarSelectComboBoxItem2**

Selects the specified page combo box item.

**Functions detail****FRStatusBarAddWndEx****Syntax**

```
void FRStatusBarAddWndEx (
    void* pParentWnd,
    FR\_StatusBarWndExCallbacks callbacks,
    FRStatusBarLocation nLocation
);
```

**Description**

Adds a windows to the status bar.

**Parameter**

pParentWnd	[In] The parent window passed from <a href="#">PILoadStatusBarUI</a> .
------------	--

callbacks	[In] The callback used to create the status bar window.
-----------	---

nLocation	[In] Specifies the location in the status bar.
-----------	--

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6032

**Related method****Since**

[SDK LATEEST VERSION > 7.1](#)

## FRStatusBarReset

### Syntax

```
void FRStatusBarReset (void );
```

### Description

Resets the status bar.

### Return

void.

### Head file reference

fr\_barTempl.h: 6045

### Since

[SDK LATEST VERSION > 7.3.1](#)

## FRStatusBarSetZoomSliderRange

### Syntax

```
void FRStatusBarSetZoomSliderRange (
    FS_INT32 nMin,
    FS_INT32 nMax
);
```

### Description

Sets the zoom slider range.

### Parameter

---

nMin	[In] The minimum value of the zoom slider range.
------	--

---

nMax	[In] The maximum value of the zoom slider range.
------	--

---

### Return

void.

### Head file reference

fr\_barTempl.h: 6054

### Since

[SDK LATEST VERSION > 7.3.1](#)

## FRStatusBarSetZoomSliderPos

### Syntax

```
void FRStatusBarSetZoomSliderPos (
```

```
FS_INT32 nPos  
);
```

**Description**

Sets the position of the zoom slider.

**Parameter**

---

nPos	[In] The position of the zoom slider.
------	---------------------------------------

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6065

**Since**

[SDK LATEEST VERSION > 7.3.1](#)

**FRStatusBarGetZoomSliderPos****Syntax**

```
FS_INT32 FRStatusBarGetZoomSliderPos (void );
```

**Description**

Gets the position of the zoom slider.

**Return**

The position of the zoom slider.

**Head file reference**

fr\_barTempl.h: 6075

**Since**

[SDK LATEEST VERSION > 7.3.1](#)

**FRStatusBarSetZoomPaneText****Syntax**

```
void FRStatusBarSetZoomPaneText (  
    FS_LPCWSTR lpwsZoomPaneText  
) ;
```

**Description**

Sets the text of the zoom pane.

**Parameter**

---

lpwsZoomPaneText	[In] The text of the zoom pane.
------------------	---------------------------------

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6084

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRStatusBarSetComboBoxPageCount****Syntax**

```
void FRStatusBarSetComboBoxPageCount (
    FS\_INT32 nCount
);
```

**Description**

Sets the page count of the page combo box.

**Parameter**

---

nCount	[In] The page count of the page combo box.
--------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6094

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FRStatusBarSelectComboBoxItem****Syntax**

```
void FRStatusBarSelectComboBoxItem (
    FS\_INT32 nIndex
);
```

**Description**

Selects the specified page combo box item.

**Parameter**

---

nIndex	[In] The index of the page combo box item.
--------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6104

**Since**

[SDK LATEST VERSION > 7.3.1](#)

**FRStatusBarGetComboBoxPageIndex****Syntax**

```
FS_INT32 FRStatusBarGetComboBoxPageIndex (void );
```

**Description**

Gets the page index of the page combo box.

**Return**

The page index of the page combo box.

**Head file reference**

fr\_barTempl.h: 6114

**Since**

[SDK LATEST VERSION > 7.3.1](#)

**FRStatusBarGetComboBoxPageText****Syntax**

```
void FRStatusBarGetComboBoxPageText (
    FS\_WideString* outText
);
```

**Description**

Gets the text of the page combo box.

**Parameter**

---

outText	[Out] It receives the text of the page combo box.
---------	---

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6123

**Since**

[SDK LATEEST VERSION > 7.3.1](#)

**FRStatusBarGetBkGroundColor****Syntax**

```
FS_COLORREF FRStatusBarGetBkGroundColor (void );
```

**Description**

Gets the back ground color of status bar in ribbon mode.

**Return**

The back ground color of status bar.

**Head file reference**

fr\_barTempl.h: 6133

**Since**

[SDK LATEEST VERSION > 7.3.1](#)

**FRStatusBarGetBkGroundPath****Syntax**

```
void FRStatusBarGetBkGroundPath (
    FS\_WideString* outPath
);
```

**Description**

Gets the path of the back ground color picture of the status bar in classic mode.

**Parameter**

---

outPath	[Out] It receives the path of the back ground color picture of the status bar in classic mode.
---------	--

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6142

**Since**

[SDK LATEEST VERSION > 7.3.1](#)

**FRStatusBarRecalcLayout**

**Syntax**

```
void FRStatusBarRecalcLayout (void );
```

**Description**

Recalculates the layout of the status bar.

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6152

**Since**

[SDK LATEEST VERSION > 8.1.0.0](#)

**FRStatusBarSetComboBoxItemCount2****Syntax**

```
void FRStatusBarSetComboBoxItemCount2 (
    FS_INT32 nCount,
    HWND hMainframeWnd
);
```

**Description**

Sets the page count of the page combo box.

**Parameter**

---

nCount	[In] The page count of the page combo box.
--------	--

---

hMainframeWnd	[In] The current mainframe handle.
---------------	------------------------------------

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6161

**Since**

[SDK LATEEST VERSION > 9.4.0](#)

**FRStatusBarSelectComboBoxItem2****Syntax**

```
void FRStatusBarSelectComboBoxItem2 (
    FS_INT32 nIndex,
    HWND hMainframeWnd
```

);

**Description**

Selects the specified page combo box item.

**Parameter**

nIndex	[In] The index of the page combo box item.
--------	--

hMainframeWnd	[In] The current mainframe handle.
---------------	------------------------------------

**Return**

void.

**Head file reference**

fr\_barTempl.h: 6172

**Since**

[SDK\\_LATEEST\\_VERSION > 9.4.0](#)

## FR\_Sys

### Description

It is used to manage the APIs of system-dependent. See [FRSysLoadStandarCursor](#) , [FRSysGetCursor](#) , [FRSysSetCursor](#) .

### Functions

#### Functions summary

**[FRSysGetCursor](#)**

Gets the current cursor.

**[FRSysGetSkinCount](#)**

Gets the count of skins for Foxit Reader.

**[FRSysGetSkinNameByIndex](#)**

Gets the skin name by index.

**[FRSysInstallDialogScrollBar](#)**

Installs the dialog scroll bar.

**[FRSysInstallDialogSkinTheme](#)**

The dialog can install the skin theme of *Foxit Reader* to keep the same. Invokes this interface in the *CReader\_Dialog::OnInitDialog* routine of the dialog.

**[FRSysLoadStandarCursor](#)**

Loads a standard cursor by type.

**[FRSysSetActiveSkinByIndex](#)**

Sets the active skin by index.

**[FRSysSetActiveSkinByName](#)**

Sets the active skin by name.

**[FRSysSetCursor](#)**

Sets the current cursor.

**[FRSysShowMessageBox](#)**

Creates, displays, and operates a message box. The message box contains an application-defined message and title, plus any combination of predefined icons and push buttons.

## Functions detail

### [FRSysGetCursor](#)

**Description**

Gets the current cursor.

**Return**

The current cursor.

**Head file reference**

fr\_sysTempl.h: 27

**Related method**

[FRSysSetCursor](#)

[FRSysLoadStandarCursor](#)

### [FRSysGetSkinCount](#)

**Syntax**

FS\_INT32 FRSysGetSkinCount (void );

**Description**

Gets the count of skins for Foxit Reader.

**Return**

The count of skins for Foxit Reader.

**Head file reference**

fr\_sysTempl.h: 88

**Related method**

**Since**

[SDK LATEEST VERSION > 8.0.0.0](#)

### [FRSysGetSkinNameByIndex](#)

**Syntax**

```
void FRSysGetSkinNameByIndex (
    FS\_INT32 nIndex,
    FS\_WideString* outName
);
```

**Description**

Gets the skin name by index.

**Parameter**

nIndex	[In] The input index.
outName	[Out] It receives the name of the skin.

**Return**

void.

**Head file reference**

fr\_sysTempl.h: 98

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 8.0.0.0](#)

**FRSysInstallDialogScrollBar****Syntax**

```
FS_BOOL FRSysInstallDialogScrollBar (
    HWND hWnd
);
```

**Description**

Installs the dialog scroll bar.

**Parameter**

hWnd	[In] The dialog to install the scroll bar.
------	--

**Return**

[True](#) means successful, otherwise not.

**Head file reference**

fr\_sysTempl.h: 77

**Related method**

**Since**  
[SDK LATEEST VERSION > 2.0](#)

### FRSysInstallDialogSkinTheme

#### Syntax

```
FS_BOOL FRSysInstallDialogSkinTheme (
    HWND hWnd
);
```

#### Description

The dialog can install the skin theme of *Foxit Reader* to keep the same. Invokes this interface in the *CReader\_Dialog::OnInitDialog* routine of the dialog.

#### Parameter

---

hWnd	[In] The dialog to install skin theme of <i>Foxit Reader</i> .
------	--

---

#### Return

[True](#) means successful, otherwise not.

#### Head file reference

fr\_sysTempl.h: 67

### FRSysLoadStandarCursor

#### Syntax

```
FR_Cursor FRSysLoadStandarCursor (
    FS_INT32 nCursorType
);
```

#### Description

Loads a standard cursor by type.

#### Parameter

---

nCursorType	[In] The type of the cursor. See <a href="#">FRCursorTypeFlags</a> .
-------------	--

---

#### Return

The specified cursor loaded.

#### Head file reference

fr\_sysTempl.h: 21

#### Related method

[FRSysSetCursor](#)

## [FRSysGetCursor](#)

**FRSysSetActiveSkinByIndex**

### Syntax

```
FS_BOOL FRSysSetActiveSkinByIndex (
    FS_INT32 nIndex
);
```

### Description

Sets the active skin by index.

### Parameter

---

nIndex	[In] The input index.
--------	-----------------------

---

### Return

TRUE for success, otherwise failure.

### Head file reference

fr\_sysTempl.h: 110

### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 8.0.0.0](#)

**FRSysSetActiveSkinByName**

### Syntax

```
FS_BOOL FRSysSetActiveSkinByName (
    nIndex
);
```

### Description

Sets the active skin by name.

### Parameter

---

nIndex	[In] The input name of the skin.
--------	----------------------------------

---

### Return

TRUE for success, otherwise failure.

### Head file reference

fr\_sysTempl.h: 121

**Related method****Since**[SDK LATEEST VERSION > 8.0.0.0](#)**FRSysSetCursor****Syntax**

```
void FRSysSetCursor (
    FR\_Cursor cursor
);
```

**Description**

Sets the current cursor.

**Parameter**

---

cursor	[In] The input cursor to be set.
--------	----------------------------------

---

**Return**

void

**Head file reference**

fr\_sysTempl.h: 26

**Related method**[FRSysGetCursor](#)[FRSysLoadStandarCursor](#)**FRSysShowMessageBox****Syntax**

```
FS_INT32 FRSysShowMessageBox (
    FS\_LPCWSTR wszPrompt,
    unsigned int nType,
    FS\_LPCWSTR wszCaption,
    FR\_MSGBOX\_CHECKBOXPARAMS* pCheckBoxParams,
    HWND hParent
);
```

**Description**

Creates, displays, and operates a message box. The message box contains an application-defined message and title, plus any combination of predefined icons and push buttons.

**Parameter**

---

wszPrompt	[In] Pointer to a null-terminated string that contains the message to be displayed.
-----------	---

---

---

nType	[In] Specifies the contents and behavior of the dialog box. See the description about <i>MessageBox</i> from MSDN.
wszCaption	[In] Pointer to a null-terminated string that contains the dialog box title. If this parameter is NULL, the default title is used.
pCheckBoxParams	[In] Pointer to a data structure representing the params of the check box on the message box. If this parameter is NULL, the default value is used.
hParent	[In] The parent window.

---

**Return**

See the description about *MessageBox* from *MSDN*.

**Head file reference**

fr\_sysTempl.h: 53

## FR\_TabBand

**[Return from](#)** **[Used by](#)**

### Description

A [FR\\_TabBand](#) is a band where documents' tabs are shown. A tab is associated with a window.

### Returned from

[FRTabBandGet](#)  
[FRTabBandGet2](#)

### Used by

[FRTabBandCloseTabWnd](#)  
[FRTabBandGetActiveTab](#)  
[FRTabBandGetActiveTabWnd](#)  
[FRTabBandGetTabsNum](#)  
[FRTabBandGetTabWnd](#)  
[FRTabBandRegisterAddBtnHandler](#)  
[FRTabBandSetActiveTab](#)

### Callbacks

#### Callbacks summary

**[FRTabBandOnClickBtn](#)**

A callback for adding button to the tab band.

**FRTabBandGetTooltip**

A callback for adding button to the tab band.

**FRTabBandTopPriority**

A callback for adding button to the tab band.

## Callbacks detail

### FRTabBandOnClickBtn

**Syntax**

```
typedef void (*FRTabBandOnClickBtn)(  
    FS_LPVVOID clientData  
)
```

**Description**

A callback for adding button to the tab band. It is invoked when the user clicks the button.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 263

**Group**

[FR\\_TabBandAddBtnCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

### FRTabBandGetTooltip

**Syntax**

```
typedef void (*FRTabBandGetTooltip)(  
    FS_LPVVOID clientData,  
    FS_WideString outTooltip  
)
```

**Description**

A callback for adding button to the tab band. It is invoked by Foxit Reader to receive the tooltip.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

outTooltip	[Out] It receives the tooltip.
------------	--------------------------------

---

**Return**

void

**Head file reference**

fr\_barExpT.h: 278

**Group**

[FR\\_TabBandAddBtnCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

## FRTabBandTopPriority

**Syntax**

```
typedef FS_BOOL (*FRTabBandTopPriority)(  
    FS_LPVVOID clientData  
)
```

**Description**

A callback for adding button to the tab band. It is invoked by Foxit Reader to determine whether this callback set is the most top priority, because only one suchbutton can be added to the tab band.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

TRUE to make the callback set as the most top priority, otherwise not.

**Head file reference**

fr\_barExpT.h: 293

**Group**

[FR\\_TabBandAddBtnCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

## Functions

### Functions summary

[FRTabBandCloseTabWnd](#)

Closes the specified tab window.

#### **FRTabBandGet**

Gets the tab band. The tab band is used to place and manage the document tabs. A tab is associated with a window.

#### **FRTabBandGet2**

Gets the tab band. The tab band is used to place and manage the document tabs. A tab is associated with a window.

#### **FRTabBandGetActiveTab**

Gets the index of the active tab window.

#### **FRTabBandGetActiveTabWnd**

Gets the window handle associated with the active tab.

#### **FRTabBandGetTabsNum**

Gets the numbers of the tabs.

#### **FRTabBandGetTabWnd**

Gets the window handle associated with the tab.

#### **FRTabBandRegisterAddBtnHandler**

Registers the callbacks for adding button to the tab band.

#### **FRTabBandSetActiveTab**

Sets the active tab window.

#### **FRTabBandSetTabIcon**

Sets the specified tab icon.

#### **FRTabBandSetTabTitle**

Sets the specified tab title.

## Functions detail

### **FRTabBandCloseTabWnd**

#### **Description**

Closes the specified tab window.

#### **Parameter**

tabBand	[In] The input tab band.
---------	--------------------------

hTabWnd	[In] The specified window handle.
---------	-----------------------------------

#### **Return**

void

#### **Head file reference**

fr\_barTempl.h: 875

**Related method****Since**[SDK LATEEST VERSION > 2.0](#)**FRTabBandGet****Syntax**

```
FR_TabBand FRTabBandGet (void );
```

**Description**

Gets the tab band. The tab band is used to place and manage the document tabs. A tab is associated with a window.

**Return**

The tab band.

**Head file reference**

fr\_barTempl.h: 832

**Since**[SDK LATEEST VERSION > 1.0](#)**FRTabBandGet2****Syntax**

```
FR_TabBand FRTabBandGet2 (
    HWND hParent
);
```

**Description**

Gets the tab band. The tab band is used to place and manage the document tabs. A tab is associated with a window.

**Parameter**

---

hParent	[In] The input mainframe window associated with the tab.
---------	--

---

**Return**

The tab band.

**Head file reference**

fr\_barTempl.h: 942

**Since**[SDK LATEEST VERSION > 1.0](#)

## FRTabBandGetActiveTab

### Syntax

```
FS_INT32 FRTabBandGetActiveTab (
    FR\_TabBand tabBand
);
```

### Description

Gets the index of the active tab window.

### Parameter

---

tabBand	[In] The input tab band.
---------	--------------------------

---

### Return

The index of the active tab window.

### Head file reference

fr\_barTempl.h: 848

### Related method

#### Since

[SDK\\_LATEEST\\_VERSION > 2.0](#)

## FRTabBandGetActiveTabWnd

### Syntax

```
HWND FRTabBandGetActiveTabWnd (
    FR\_TabBand tabBand
);
```

### Description

Gets the window handle associated with the active tab.

### Parameter

---

tabBand	[In] The input tab band.
---------	--------------------------

---

### Return

The window handle associated with the active tab.

### Head file reference

fr\_barTempl.h: 848

### Related method

#### Since

## [SDK\\_LATEEST\\_VERSION > 2.0](#)

### FRTabBandGetTabsNum

#### Syntax

```
FS_INT32 FRTabBandGetTabsNum (
    FR\_TabBand tabBand
);
```

#### Description

Gets the numbers of the tabs.

#### Parameter

---

tabBand	[In] The input tab band.
---------	--------------------------

---

#### Return

The numbers of the tabs.

#### Head file reference

fr\_barTempl.h: 864

#### Related method

#### Since

## [SDK\\_LATEEST\\_VERSION > 2.0](#)

### FRTabBandGetTabWnd

#### Syntax

```
HWND FRTabBandGetTabWnd (
    FR\_TabBand tabBand,
    FS\_INT32 iTab
);
```

#### Description

Gets the window handle associated with the tab.

#### Parameter

---

tabBand	[In] The input tab band.
---------	--------------------------

---

iTab	[In] The specified index.
------	---------------------------

---

#### Return

The window handle associated with the tab.

**Head file reference**

fr\_barTempl.h: 841

**Related method**[FRTabBandGetActiveTabWnd](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.0](#)**FRTabBandRegisterAddBtnHandler****Syntax**

```
FS_BOOL FRTabBandRegisterAddBtnHandler (
    FR\_TabBandAddBtnCallbacks callbacks
);
```

**Description**

Registers the callbacks for adding button to the tab band.

**Parameter**

---

callbacks	[In] The input callbacks for adding button to the tab band.
-----------	---

---

**Return**[TRUE](#) if success, otherwise failure.**Head file reference**

fr\_barTempl.h: 910

**Since**[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)**FRTabBandSetActiveTab****Syntax**

```
FS_BOOL FRTabBandSetActiveTab (
    FR\_TabBand tabBand,
    HWND hTabWnd
);
```

**Description**

Sets the active tab window.

**Parameter**

---

tabBand	[In] The input tab band.
---------	--------------------------

---

---

<code>hTabWnd</code>	[In] The specified tab window.
----------------------	--------------------------------

---

**Return**TRUE if success, otherwise failure.**Head file reference**

fr\_barTempl.h: 898

**Related method**[FRTabBandGetActiveTab](#)**Since**[SDK LATEEST VERSION > 2.0](#)**FRTabBandSetTabIcon****Syntax**

```
void FRTabBandSetTabIcon (
    HWND hChildFrame,
    HICON hIcon
);
```

**Description**

Sets the specified tab icon.

**Parameter**


---

<code>hChildFrame</code>	[In] The input child frame window associated with the tab.
--------------------------	--

---

<code>hIcon</code>	[In] The input icon set to the tab.
--------------------	-------------------------------------

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 931

**Since**[SDK LATEEST VERSION > 7.3.1.0](#)**FRTabBandSetTabTitle****Syntax**

```
FS_BOOL FRTabBandSetTabTitle (
    HWND hChildHwnd,
    FS_LPCWSTR lpwsTitle
);
```

**Description**

Sets the specified tab title.

**Parameter**

---

hChildHwnd	[In] The input child frame window associated with the tab.
------------	--

---

lpwsTitle	[In] The input title.
-----------	-----------------------

---

**Return**

[TRUE](#) if success, otherwise failure.

**Head file reference**

fr\_barTempl.h: 920

**Since**

[SDK LATEEST VERSION > 7.3.1.0](#)

## FR\_TextSelectTool

[Return from Used by](#)

### Description

The text select tool is used to process the operation of text selecting in the doc view.

### Returned from

[FRDocGetTextSelectTool](#)

[FRTextSelectToolCreate](#)

### Used by

[FRTextSelectToolClearSelection](#)

[FRTextSelectToolDestroyTextSelectTool](#)

[FRTextSelectToolDoLButtonDblClk](#)

[FRTextSelectToolDoLButtonDown](#)

[FRTextSelectToolDoLButtonUp](#)

[FRTextSelectToolDoMouseMove](#)

[FRTextSelectToolDoMouseWheel](#)

[FRTextSelectToolDoRButtonUp](#)

[FRTextSelectToolDrawSelection](#)

[FRTextSelectToolEnumTextObjectRect](#)

[FRTextSelectToolGetSelectedText](#)

[FRTextSelectToolIsSelecting](#)

### Functions

## Functions summary

### [\*\*FRTTextSelectToolClearSelection\*\*](#)

Clears the area where the text is selected.

### [\*\*FRTTextSelectToolCreate\*\*](#)

Creates the text select tool which is used to process the operation of text selecting in the doc view.

### [\*\*FRTTextSelectToolDestroyTextSelectTool\*\*](#)

Destroys the text select tool created from [FRTTextSelectToolCreate](#) .

### [\*\*FRTTextSelectToolDoLButtonDblClick\*\*](#)

Does the mouse left button double-click operation.

### [\*\*FRTTextSelectToolDoLButtonDown\*\*](#)

Does the mouse left button down operation.

### [\*\*FRTTextSelectToolDoLButtonUp\*\*](#)

Does the mouse left button up operation.

### [\*\*FRTTextSelectToolDoMouseMove\*\*](#)

Does the mouse move operation.

### [\*\*FRTTextSelectToolDoMouseWheel\*\*](#)

Does the mouse wheel operation.

### [\*\*FRTTextSelectToolDoRButtonUp\*\*](#)

Does the mouse right button up operation.

### [\*\*FRTTextSelectToolDrawSelection\*\*](#)

Draws the area where the text is selected in black color.

### [\*\*FRTTextSelectToolEnumTextObjectRect\*\*](#)

Enumerates the text object rectangles. Invokes this interface first to get the count of the rectangles by setting *outRectArray* as [NULL](#) . Then allocates the buffer for the rectangle array.

### [\*\*FRTTextSelectToolGetSelectedText\*\*](#)

Gets the selected text.

### [\*\*FRTTextSelectToolIsSelecting\*\*](#)

Checks whether the mouse is selecting text.

## Functions detail

### FRTTextSelectToolClearSelection

#### Syntax

```
void FRTTextSelectToolClearSelection (
    FR\_TextSelectTool textSelectTool
);
```

#### Description

Clears the area where the text is selected.

#### Parameter

---

textSelectTool	[In] The input text select tool.
----------------	----------------------------------

---

#### Return

**Head file reference**

fr\_viewTempl.h: 839

**Related method****FRTTextSelectToolCreate****Syntax**

```
FR_TextSelectTool FRTTextSelectToolCreate (
    FR Document doc
);
```

**Description**

Creates the text select tool which is used to process the operation of text selecting in the doc view.

**Parameter**

---

doc	[In] The input doc where the operation of text selecting occurs.
-----	--

---

**Return**

The text select tool.

**Head file reference**

fr\_viewTempl.h: 796

**Related method****FRTTextSelectToolDestroyTextSelectTool****Syntax**

```
void FRTTextSelectToolDestroyTextSelectTool (
    FR TextSelectTool textSelectTool
);
```

**Description**

Destroys the text select tool created from [FRTTextSelectToolCreate](#).

**Parameter**

---

textSelectTool	[In] The input text select tool.
----------------	----------------------------------

---

**Return****Head file reference**

fr\_viewTempl.h: 806

**Related method****FRTTextSelectToolDoLButtonDblClk****Syntax**

```
void FRTTextSelectToolDoLButtonDblClk (
    FR\_TextSelectTool textSelectTool,
    FR\_PageView pageView,
    FS\_DevicePoint point
);
```

**Description**

Does the mouse left button double-click operation.

**Parameter**

textSelectTool	[In] The input text select tool.
pageView	[In] The input page view where do the operation of text selecting.
point	[In] The input point where do the mouse operation.

**Return****Head file reference**

fr\_viewTempl.h: 873

**Related method****FRTTextSelectToolDoLButtonDown****Syntax**

```
void FRTTextSelectToolDoLButtonDown (
    FR\_TextSelectTool textSelectTool,
    FR\_PageView pageView,
    FS\_DevicePoint point
);
```

**Description**

Does the mouse left button down operation.

**Parameter**

textSelectTool	[In] The input text select tool.
----------------	----------------------------------

---

pageView	[In] The input page view where do the operation of text selecting.
----------	--

---

point	[In] The input point where do the mouse operation.
-------	--

---

**Return****Head file reference**

fr\_viewTempl.h: 861

**Related method****FRTTextSelectToolDoLButtonUp****Syntax**

```
void FRTTextSelectToolDoLButtonUp (
    FR\_TextSelectTool textSelectTool,
    FR\_PageView pageView,
    FS\_DevicePoint point
);
```

**Description**

Does the mouse left button up operation.

**Parameter**


---

textSelectTool	[In] The input text select tool.
----------------	----------------------------------

---

pageView	[In] The input page view where do the operation of text selecting.
----------	--

---

point	[In] The input point where do the mouse left button up operation.
-------	---

---

**Return****Head file reference**

fr\_viewTempl.h: 849

**Related method****FRTTextSelectToolDoMouseMove****Syntax**

```
void FRTTextSelectToolDoMouseMove (
    FR\_TextSelectTool textSelectTool,
    FR\_PageView pageView,
```

---

```
FS_DevicePoint point
);
```

**Description**

Does the mouse move operation.

**Parameter**


---

textSelectTool	[In] The input text select tool.
pageView	[In] The input page view where do the operation of text selecting.
point	[In] The input point where do the mouse operation.

---

**Return****Head file reference**

fr\_viewTempl.h: 897

**Related method****FRTTextSelectToolDoMouseWheel****Syntax**

```
FS_BOOL FRTTextSelectToolDoMouseWheel (
    FR_TextSelectTool textSelectTool,
    FR_PageView pageView,
    FS_DevicePoint point
);
```

**Description**

Does the mouse wheel operation.

**Parameter**


---

textSelectTool	[In] The input text select tool.
pageView	[In] The input page view where do the operation of text selecting.
point	[In] The input point where do the mouse operation.

---

**Return**

TRUE for success, otherwise not.

**Head file reference**

fr\_viewTempl.h: 909

**Related method****FRTTextSelectToolDoRButtonUp****Syntax**

```
void FRTTextSelectToolDoRButtonUp (
    FR\_TextSelectTool textSelectTool,
    FR\_PageView pageView,
    FS\_DevicePoint point
);
```

**Description**

Does the mouse right button up operation.

**Parameter**

textSelectTool	[In] The input text select tool.
pageView	[In] The input page view where do the operation of text selecting.
point	[In] The input point where do the mouse operation.

**Return****Head file reference**

fr\_viewTempl.h: 885

**Related method****FRTTextSelectToolDrawSelection****Syntax**

```
void FRTTextSelectToolDrawSelection (
    FR\_TextSelectTool textSelectTool,
    FR\_DocView docView,
    FR\_WinPort winPort
);
```

**Description**

Draws the area where the text is selected in black color.

**Parameter**

---

textSelectTool	[In] The input text select tool.
docView	[In] The input doc view where do the operation of text selecting.
winPort	[In] The platform-depend things.

---

**Return****Head file reference**

fr\_viewTempl.h: 827

**Related method****FRTTextSelectToolEnumTextObjectRect****Syntax**

```
FS_BOOL FRTTextSelectToolEnumTextObjectRect (
    FR\_TextSelectTool textSelectTool,
    FS_INT32 pageIndex,
    FS\_FloatRect** outRectArray,
    FS_INT32* nCount
);
```

**Description**

Enumerates the text object rectangles. Invokes this interface first to get the count of the rectangles by setting *outRectArray* as [NULL](#). Then allocates the buffer for the rectangle array.

**Parameter**


---

textSelectTool	[In] The input text select tool.
pageIndex	[In] The input page index.
outRectArray	[Out] It receives the rectangle array of text object.
nCount	[Out] It receives the count of the rectangle array.

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_viewTempl.h: 931

**Related method**

## FRTTextSelectToolGetSelectedText

### Syntax

```
FS_BOOL FRTTextSelectToolGetSelectedText (
    FR\_TextSelectTool textSelectTool,
    FS\_WideString* outText
);
```

### Description

Gets the selected text.

### Parameter

---

textSelectTool	[In] The input text select tool.
----------------	----------------------------------

---

outText	[Out] It receives the text selected.
---------	--------------------------------------

---

### Return

[TRUE](#) means successful, otherwise not.

### Head file reference

fr\_viewTempl.h: 816

### Related method

## FRTTextSelectToolIsSelecting

### Syntax

```
FS_BOOL FRTTextSelectToolIsSelecting (
    FR\_TextSelectTool textSelectTool
);
```

### Description

Checks whether the mouse is selecting text.

### Parameter

---

textSelectTool	[In] The input text select tool.
----------------	----------------------------------

---

### Return

[TRUE](#) means the mouse is selecting text.

### Head file reference

fr\_viewTempl.h: 921

**Related method**

## FR\_ThumbnailView

**[Return from Used by](#)**

### Description

A [FR\\_ThumbnailView](#) is a view thumbnail view related to the document view. It displays the thumbnail of the document in the page panel.

### Returned from

**[FRDocViewGetThumbnailView](#)**

### Used by

[FRThumbnailViewCountPage](#)  
[FRThumbnailViewCountVisiblePage](#)  
[FRThumbnailViewGetPageRect](#)  
[FRThumbnailViewGetPDPAGE](#)  
[FRThumbnailViewGetVisiblePageRange](#)

## Functions

### Functions summary

**[FRThumbnailViewCountPage](#)**

Gets the count of page displayed in the thumbnail view.

**[FRThumbnailViewCountVisiblePage](#)**

Gets the count of visible page displayed in the thumbnail view.

**[FRThumbnailViewGetPageRect](#)**

Gets the rectangle of specified visible page.

**[FRThumbnailViewGetPDPAGE](#)**

Gets a [FPD\\_Page](#) object.

**[FRThumbnailViewGetVisiblePageRange](#)**

Gets the index of visible page range.

### Functions detail

#### **FRThumbnailViewCountPage**

**Syntax**

```
FS_INT32 FRThumbnailViewCountPage (
    FR\_ThumbnailView tv
);
```

**Description**

Gets the count of page displayed in the thumbnail view.

**Parameter**

---

tv	[In] The input thumbnail view.
----	--------------------------------

---

**Return**

The count of page displayed in the thumbnail view.

**Head file reference**

fr\_viewTempl.h: 955

**Related method**

[FRDocViewGetThumbnailView](#)

**FRThumbnailViewCountVisiblePage****Syntax**

```
FS_INT32 FRThumbnailViewCountVisiblePage (
    FR\_ThumbnailView tv
);
```

**Description**

Gets the count of visible page displayed in the thumbnail view.

**Parameter**

---

tv	[In] The input thumbnail view.
----	--------------------------------

---

**Return**

The count of visible page displayed in the thumbnail view.

**Head file reference**

fr\_viewTempl.h: 965

**Related method**

[FRDocViewGetThumbnailView](#)

**FRThumbnailViewGetPageRect****Syntax**

```
FS_Rect FRThumbnailViewGetPageRect (
    FR\_ThumbnailView tv,
    FS_INT32 nPage
);
```

**Description**

Gets the rectangle of specified visible page.

**Parameter**

---

tv	[In] The input thumbnail view.
----	--------------------------------

---

---

nPage	[In] The index of visible pages.
-------	----------------------------------

---

**Return**

The rectangle of specified visible page.

**Head file reference**

fr\_viewTempl.h: 975

**Related method**

[FRDocViewGetThumbnailView](#)

**FRThumbnailViewGetPDPAGE****Syntax**

```
FPD_Page FRThumbnailViewGetPDPAGE (
    FR\_ThumbnailView tv,
    FS\_INT32 nPage
);
```

**Description**

Gets a [FPD\\_Page](#) object.

**Parameter**

---

tv	[In] The input thumbnail view.
----	--------------------------------

---

---

nPage	[In] The index of page.
-------	-------------------------

---

**Return**

The [FPD\\_Page](#) object.

**Head file reference**

fr\_viewTempl.h: 998

**Related method**

[FRDocViewGetThumbnailView](#)

**FRThumbnailViewGetVisiblePageRange****Syntax**

```
void FRThumbnailViewGetVisiblePageRange (
```

---

```
FR_ThumbnailView tv,
FS_INT32* nFrom,
FS_INT32* nTo
);
```

**Description**

Gets the index of visible page range.

**Parameter**

tv	[In] The input thumbnail view.
nFrom	[Out] It receives the starting index of visible pages.
nTo	[Out] It receives the ending index of visible pages.

**Return**

void

**Head file reference**

fr\_viewTempl.h: 986

**Related method**

[FRDocViewGetThumbnailView](#)

## FR\_Tool

[Return from](#) [Used by](#)**Description**

A [FR\\_Tool](#) is an object that can handle key presses and mouse events in the region of a document view. Tools do not handle mouse events in other parts of Foxit Reader window, such as navigation pane. At any time, there is only one active tool, which a plug-in can set using [FRAppSetActiveTool](#) ().

Reader has some built-in tools, such as Hand tool, Zoom tool, Link tool. You can get other built-in tools using [FRAppGetToolByName](#) () with the tool's name defined in the [BuildInToolName](#) group.

Use [FRAppRegisterTool](#) () to add a new tool to Foxit Reader.

**Returned from**

[FRAppGetActiveTool](#)  
[FRAppGetToolBar](#)  
[FRAppGetToolBarByName](#)  
[FRAppGetToolByIndex](#)  
[FRAppGetToolByName](#)  
[FRToolBarGetButton](#)

[FRToolBarGetButtonByName](#)  
[FRToolBarNew](#)  
[FRToolBarNewFlyout](#)  
[FRToolButtonNew](#)  
[FRToolNew](#)

Used by

[FRAppRegisterTool](#)  
[FRAppSetActiveTool](#)  
[FRToolBarAddButton](#)  
[FRToolBarCountButtons](#)  
[FRToolBarDock](#)  
[FRToolBarGetButton](#)  
[FRToolBarGetButtonByName](#)  
[FRToolBarGetName](#)  
[FRToolBarGetTitle](#)  
[FRToolBarHideButtonInBrowser](#)  
[FRToolBarHideToolBar](#)  
[FRToolBarHideToolBar](#)  
[FRToolBarInsertButton](#)  
[FRToolBarIsDisable](#)  
[FRToolBarIsFlyOutToolbar](#)  
[FRToolBarIsVisible](#)  
[FRToolBarRelease](#)  
[FRToolBarRemoveButton](#)  
[FRToolBarSetDefaultToolbar](#)  
[FRToolBarSetDisable](#)  
[FRToolBarSetTitle](#)  
[FRToolBarSetTitle](#)  
[FRToolBarShowToolBar](#)  
[FRToolBarShowToolBar](#)  
[FRToolBarUpdateButtonStates](#)  
[FRToolButtonExecuteProc](#)  
[FRToolButtonGetClientData](#)  
[FRToolButtonGetLabelText](#)  
[FRToolButtonGetMapId](#)  
[FRToolButtonGetName](#)  
[FRToolButtonIsSeparator](#)  
[FRToolButtonIsVisible](#)  
[FRToolButtonRelease](#)  
[FRToolButtonSetClientData](#)  
[FRToolButtonSetDropDownProc](#)  
[FRToolButtonSetEnableProc](#)  
[FRToolButtonSetExecuteProc](#)  
[FRToolButtonSetFlyoutToolBar](#)  
[FRToolButtonSetHelpText](#)  
[FRToolButtonSetIcon](#)  
[FRToolButtonSetLabelText](#)  
[FRToolButtonSetMarkedProc](#)  
[FRToolGetName](#)  
[FRToolNew](#)  
[FRToolRelease](#)  
[FRToolSetAssociatedMousePtHandlerType](#)  
[FRToolSetAssociatedSelectionHandlerType](#)

## Definitions

### Definitions summary

#### **FR\_NAME\_ADVANCEDSEARCHPAGE**

The name of advance search tool

#### **FR\_NAME\_ANNOT**

The name of Annot Tool

#### **FR\_NAME\_AREA**

The name of Area Tool

#### **FR\_NAME\_ARROW**

The name of Arrow Tool

#### **FR\_NAME\_CALLOUT**

The name of Callout

#### **FR\_NAME\_CARET**

The name of Insert Text Tool

#### **FR\_NAME\_CHECKBOX**

The name of Check Box Tool

#### **FR\_NAME\_CIRCLE**

The name of Circle Tool

#### **FR\_NAME\_CLOUDY**

The name of Cloudy Tool

#### **FR\_NAME\_COMBOBOX**

The name of Combo Box Tool

#### **FR\_NAME\_DFA\_TOOLNAME**

The name of Attach a file

#### **FR\_NAME\_DIMENSION**

The name of Dimension Tool

#### **FR\_NAME\_DISTANCE**

The name of Distance Tool

#### **FR\_NAME\_EDITSELECT**

The name of Edit Selected Text

#### **FR\_NAME\_ELLIPSE**

The name of Ellipse Tool

#### **FR\_NAME\_FAAToolNAME**

The name of FileAttachment Tool

#### **FR\_NAME\_FINDTEXT**

The name of Find Text Tool

#### **FR\_NAME\_HAND**

Hand Tool

#### **FR\_NAME\_HIGHLIGHT**

The name of Highlight Tool

#### **FR\_NAME\_LINE**

The name of Line Tool

#### **FR\_NAME\_LISTBOX**

The name of List Box Tool

#### **FR\_NAME\_LOUPETOOL**

The name of Loupe Tool

#### **FR\_NAME\_MANGIFIER**

The name of Magnifier

**FR\_NAME\_MOVIE**

The name of Movie Tool

**FR\_NAME\_NOTE**

The name of Note Tool

**FR\_NAME\_PENCIL**

The name of Pencil Tool

**FR\_NAME\_PERIMETER**

The name of Perimeter Tool

**FR\_NAME\_POLYGON**

The name of Polygon Tool

**FR\_NAME\_POLYLINE**

The name of Polyline Tool

**FR\_NAME\_PUSHBUTTON**

The name of Push Button Tool

**FR\_NAME\_QUADRILATERALLINK**

The name of Quadrilateral Link Tool

**FR\_NAME\_RADIOBUTTON**

The name of Radio Button Tool

**FR\_NAME\_RECTANGLE**

The name of Rectangle Tool

**FR\_NAME\_RECTANGLELINK**

The name of Rectangle Link Tool

**FR\_NAME\_REPLACE**

The name of Replace Tool

**FR\_NAME\_RUBBER**

The name of Rubber Tool

**FR\_NAME\_SCREEN**

The name of Image Tool

**FR\_NAME\_SELECTTEXT**

The name of Select Text Tool

**FR\_NAME\_SNAPSHOT**

The name of Snapshot Tool

**FR\_NAME\_SOUND**

The name of Sound Tool

**FR\_NAME\_SQUARE**

The name of Square Tool

**FR\_NAME\_SQUIGGLY**

The name of Squiggly Tool

**FR\_NAME\_STAMP**

The name of stamp tool

**FR\_NAME\_STRIKEOUT**

The name of Strikeout Tool

**FR\_NAME\_TEXTBOX**

The name of Textbox

**FR\_NAME\_TEXTFIELD**

The name of Text Field Tool

**FR\_NAME\_TYPEWRITER**

The name of Typewriter

**FR\_NAME\_UNDERLINE**

The name of Underline Tool

## Definitions detail

### FR\_NAME\_ADVANCEDSEARCHPAGE

#### Syntax

```
#define FR_NAME_ADVANCEDSEARCHPAGE "Search"
```

#### Description

The name of advance search tool

#### Group

[BuildInToolName](#)

#### Head file reference

fr\_appExpT.h: 655

### FR\_NAME\_ANNOT

#### Syntax

```
#define FR_NAME_ANNOT "Annot Tool"
```

#### Description

The name of Annot Tool

#### Group

[BuildInToolName](#)

#### Head file reference

fr\_appExpT.h: 529

### FR\_NAME\_AREA

#### Syntax

```
#define FR_NAME_AREA "Area Tool"
```

#### Description

The name of Area Tool

#### Group

[BuildInToolName](#)

#### Head file reference

fr\_appExpT.h: 622

### FR\_NAME\_ARROW

#### Syntax

```
#define FR_NAME_ARROW "Arrow Tool"
```

**Description**

The name of Arrow Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 538

## FR\_NAME\_CALLOUT

**Syntax**

```
#define FR_NAME_CALLOUT "Callout"
```

**Description**

The name of Callout

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 646

## FR\_NAME\_CARET

**Syntax**

```
#define FR_NAME_CARET "Insert Text Tool"
```

**Description**

The name of Insert Text Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 592

## FR\_NAME\_CHECKBOX

**Syntax**

```
#define FR_NAME_CHECKBOX "Check Box Tool"
```

**Description**

The name of Check Box Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 601

## FR\_NAME\_CIRCLE

**Syntax**

#define FR\_NAME\_CIRCLE "Circle Tool"

**Description**

The name of Circle Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 553

## FR\_NAME\_CLOUDY

**Syntax**

#define FR\_NAME\_CLOUDY "Cloudy Tool"

**Description**

The name of Cloudy Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 562

## FR\_NAME\_COMBOBOX

**Syntax**

#define FR\_NAME\_COMBOBOX "Combo Box Tool"

**Description**

The name of Combo Box Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 607

## FR\_NAME\_DFA\_TOOLNAME

### Syntax

#define FR\_NAME\_DFA\_TOOLNAME "Attach a file"

### Description

The name of Attach a file

### Group

[BuildInToolName](#)

### Head file reference

fr\_appExpT.h: 631

## FR\_NAME\_DIMENSION

### Syntax

#define FR\_NAME\_DIMENSION "Dimension Tool"

### Description

The name of Dimension Tool

### Group

[BuildInToolName](#)

### Head file reference

fr\_appExpT.h: 544

## FR\_NAME\_DISTANCE

### Syntax

#define FR\_NAME\_DISTANCE "Distance Tool"

### Description

The name of Distance Tool

### Group

[BuildInToolName](#)

### Head file reference

fr\_appExpT.h: 616

## FR\_NAME\_EDITSELECT

**Syntax**

```
#define FR_NAME_EDITSELECT "Edit Selected Text"
```

**Description**

The name of Edit Selected Text

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 574

## FR\_NAME\_ELLIPSE

**Syntax**

```
#define FR_NAME_ELLIPSE "Ellipse Tool"
```

**Description**

The name of Ellipse Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 556

## FR\_NAME\_FAATOOLNAME

**Syntax**

```
#define FR_NAME_FAATOOLNAME "FileAttachment Tool"
```

**Description**

The name of FileAttachment Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 628

## FR\_NAME\_FINDTEXT

**Syntax**

```
#define FR_NAME_FINDTEXT "Find Text Tool"
```

**Description**

The name of Find Text Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 523

## FR\_NAME\_HAND

**Syntax**

#define FR\_NAME\_HAND "Hand Tool"

**Description**

Hand Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 517

## FR\_NAME\_HIGHLIGHT

**Syntax**

#define FR\_NAME\_HIGHLIGHT "Highlight Tool"

**Description**

The name of Highlight Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 577

## FR\_NAME\_LINE

**Syntax**

#define FR\_NAME\_LINE "Line Tool"

**Description**

The name of Line Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 541

## FR\_NAME\_LISTBOX

### Syntax

#define FR\_NAME\_LISTBOX "List Box Tool"

### Description

The name of List Box Tool

### Group

[BuildInToolName](#)

### Head file reference

fr\_appExpT.h: 610

## FR\_NAME\_LOUPETOOL

### Syntax

#define FR\_NAME\_LOUPETOOL "Loupe Tool"

### Description

The name of Loupe Tool

### Group

[BuildInToolName](#)

### Head file reference

fr\_appExpT.h: 634

## FR\_NAME\_MANGIFIER

### Syntax

#define FR\_NAME\_MANGIFIER "Magnifier"

### Description

The name of Magnifier

### Group

[BuildInToolName](#)

### Head file reference

fr\_appExpT.h: 637

## FR\_NAME\_MOVIE

**Syntax**

```
#define FR_NAME_MOVIE "Movie Tool"
```

**Description**

The name of Movie Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 652

## FR\_NAME\_NOTE

**Syntax**

```
#define FR_NAME_NOTE "Note Tool"
```

**Description**

The name of Note Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 595

## FR\_NAME\_PENCIL

**Syntax**

```
#define FR_NAME_PENCIL "Pencil Tool"
```

**Description**

The name of Pencil Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 568

## FR\_NAME\_PERIMETER

**Syntax**

```
#define FR_NAME_PERIMETER "Perimeter Tool"
```

**Description**

The name of Perimeter Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 619

## FR\_NAME\_POLYGON

**Syntax**

#define FR\_NAME\_POLYGON "Polygon Tool"

**Description**

The name of Polygon Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 559

## FR\_NAME\_POLYLINE

**Syntax**

#define FR\_NAME\_POLYLINE "Polyline Tool"

**Description**

The name of Polyline Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 565

## FR\_NAME\_PUSHBUTTON

**Syntax**

#define FR\_NAME\_PUSHBUTTON "Push Button Tool"

**Description**

The name of Push Button Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 598

## FR\_NAME\_QUADRILATERALLINK

### Syntax

```
#define FR_NAME_QUADRILATERALLINK "Quadrilateral Link Tool"
```

### Description

The name of Quadrilateral Link Tool

### Group

[BuildInToolName](#)

### Head file reference

fr\_appExpT.h: 535

## FR\_NAME\_RADIOBUTTON

### Syntax

```
#define FR_NAME_RADIOBUTTON "Radio Button Tool"
```

### Description

The name of Radio Button Tool

### Group

[BuildInToolName](#)

### Head file reference

fr\_appExpT.h: 604

## FR\_NAME\_RECTANGLE

### Syntax

```
#define FR_NAME_RECTANGLE "Rectangle Tool"
```

### Description

The name of Rectangle Tool

### Group

[BuildInToolName](#)

### Head file reference

fr\_appExpT.h: 550

## FR\_NAME\_RECTANGLELINK

**Syntax**

```
#define FR_NAME_RECTANGLELINK "Rectangle Link Tool"
```

**Description**

The name of Rectangle Link Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 532

## FR\_NAME\_REPLACE

**Syntax**

```
#define FR_NAME_REPLACE "Replace Tool"
```

**Description**

The name of Replace Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 589

## FR\_NAME\_RUBBER

**Syntax**

```
#define FR_NAME_RUBBER "Rubber Tool"
```

**Description**

The name of Rubber Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 571

## FR\_NAME\_SCREEN

**Syntax**

```
#define FR_NAME_SCREEN "Image Tool"
```

**Description**

The name of Image Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 625

## FR\_NAME\_SELECTTEXT

**Syntax**

#define FR\_NAME\_SELECTTEXT "Select Text Tool"

**Description**

The name of Select Text Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 520

## FR\_NAME\_SNAPSHOT

**Syntax**

#define FR\_NAME\_SNAPSHOT "Snapshot Tool"

**Description**

The name of Snapshot Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 526

## FR\_NAME\_SOUND

**Syntax**

#define FR\_NAME\_SOUND "Sound Tool"

**Description**

The name of Sound Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 649

## FR\_NAME\_SQUARE

### Syntax

```
#define FR_NAME_SQUARE "Square Tool"
```

### Description

The name of Square Tool

### Group

[BuildInToolName](#)

### Head file reference

fr\_appExpT.h: 547

## FR\_NAME\_SQUIGGLY

### Syntax

```
#define FR_NAME_SQUIGGLY "Squiggly Tool"
```

### Description

The name of Squiggly Tool

### Group

[BuildInToolName](#)

### Head file reference

fr\_appExpT.h: 586

## FR\_NAME\_STAMP

### Syntax

```
#define FR_NAME_STAMP "Stamp"
```

### Description

The name of stamp tool

### Group

[BuildInToolName](#)

### Head file reference

fr\_appExpT.h: 658

## FR\_NAME\_STRIKEOUT

**Syntax**

```
#define FR_NAME_STRIKEOUT "Strikeout Tool"
```

**Description**

The name of Strikeout Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 583

## FR\_NAME\_TEXTBOX

**Syntax**

```
#define FR_NAME_TEXTBOX "Textbox"
```

**Description**

The name of Textbox

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 643

## FR\_NAME\_TEXTFIELD

**Syntax**

```
#define FR_NAME_TEXTFIELD "Text Field Tool"
```

**Description**

The name of Text Field Tool

**Group**

[BuildInToolName](#)

**Head file reference**

fr\_appExpT.h: 613

## FR\_NAME\_TYPEWRITER

**Syntax**

```
#define FR_NAME_TYPEWRITER "Typewriter"
```

**Description**

The name of Typewriter

**Group**[BuildInToolName](#)**Head file reference**

fr\_appExpT.h: 640

**FR\_NAME\_UNDERLINE****Syntax**

#define FR\_NAME\_UNDERLINE "Underline Tool"

**Description**

The name of Underline Tool

**Group**[BuildInToolName](#)**Head file reference**

fr\_appExpT.h: 580

## Callbacks

### Callbacks summary

[FRCmdMsgOnCmdMsgByName](#)

A callback for application level event handler.

[FRToolOnInit](#)The initialization for [FR\\_Tool](#).[FRToolDestroy](#)A callback for [FR\\_Tool](#).[FRToolOnActivate](#)A callback for [FR\\_Tool](#).[FRToolOnDeactivate](#)A callback for [FR\\_Tool](#).[FRToolOnKeyDown](#)A callback for [FR\\_Tool](#).[FRToolOnKeyUp](#)A callback for [FR\\_Tool](#).[FRToolOnChar](#)A callback for [FR\\_Tool](#).[FRToolOnLeavePage](#)A callback for [FR\\_Tool](#).[FRToolIsEnabled](#)A callback for [FR\\_Tool](#).[FRToolOnLButtonDown](#)A callback for [FR\\_Tool](#).[FRToolOnLButtonUp](#)

A callback for [FR\\_Tool](#) .

#### [\*\*FRToolOnLButtonDblClk\*\*](#)

A callback for [FR\\_Tool](#) .

#### [\*\*FRToolOnMouseMove\*\*](#)

A callback for [FR\\_Tool](#) .

#### [\*\*FRToolOnRButtonDown\*\*](#)

A callback for [FR\\_Tool](#) .

#### [\*\*FRToolOnRButtonUp\*\*](#)

A callback for [FR\\_Tool](#) .

#### [\*\*FRToolOnRButtonDblClk\*\*](#)

A callback for [FR\\_Tool](#) .

#### [\*\*FRToolOnMouseWheel\*\*](#)

A callback for [FR\\_Tool](#) .

#### [\*\*FRToolOnDraw\*\*](#)

A callback for [FR\\_Tool](#) .

#### [\*\*FRToolIsProcessing\*\*](#)

A callback for [FR\\_Tool](#) .

#### [\*\*FRToolOnMouseHover\*\*](#)

A callback for [FR\\_Tool](#) .

#### [\*\*FRToolIsWndCapturing\*\*](#)

A callback for [FR\\_Tool](#) .

## Callbacks detail

### FRCmdMsgOnCmdMsgByName

#### Syntax

```
typedef FS_BOOL (*FRCmdMsgOnCmdMsgByName)(  
    FS_LPVVOID clientData,  
    FS_LPCSTR lpsName,  
    FS_INT32 nCode,  
    void* pExtra,  
    void* pHandlerInfo  
);
```

#### Description

A callback for application level event handler. It is called by the *Foxit Reader* to route and dispatch command messages and to handle the update of command user-interface objects, such as menu, toolbar.

#### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

lpsName	[In] The name of menu or toolbar whose command messages need to be routed and dispatched.
---------	---

---

nCode	[In] References to MFC CCmdTarget::OnCmdMsg .
-------	---

---

---

pExtra	[In] References to <i>MFC CCmdTarget::OnCmdMsg</i> .
--------	--

---

pHandlerInfo	[In] It represents the <i>MFC</i> struct <i>AFX_CMDHANDLERINFO</i> .
--------------	--

---

**Return**

Nonzero if the message is handled; otherwise 0.

**Head file reference**

fr\_appExpT.h: 693

**Group**

[FR\\_CmdMsgEventCallbacksRec](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRTToolOnInit****Syntax**

```
typedef FS_BOOL (*FRTToolOnInit)(
    FS\_LPVVOID clientData
);
```

**Description**

The initialization for [FR\\_Tool](#) . It is called when a tool starts to creat.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

If [TRUE](#) , the tool will be created, if [FALSE](#) , the tool will be unloaded.

**Head file reference**

fr\_appExpT.h: 198

**Group**

[FR\\_ToolCallbacksRec](#)

**FRTToolDestroy****Syntax**

```
typedef FS_BOOL (*FRTToolDestroy)(
    FS\_LPVVOID clientData
);
```

**Description**

A callback for [FR\\_Tool](#) . It is called at shutdown time to allow the tool to free dynamic memory.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

[TRUE](#) for success, otherwise [FALSE](#) .

**Head file reference**

fr\_appExpT.h: 212

**Group**

[FR\\_ToolCallbacksRec](#)

**FRTToolOnActivate****Syntax**

```
typedef void (*FRTToolOnActivate)(  
    FS\_LPVOID clientData,  
    FS\_BOOL bPersistent  
)
```

**Description**

A callback for [FR\\_Tool](#) . It is called when a tool has become the active tool.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

bPersistent	[In] A flag indicates whether the tool should remain active.
-------------	--

---

**Return****Head file reference**

fr\_appExpT.h: 225

**Group**

[FR\\_ToolCallbacksRec](#)

**FRTToolOnDeactivate****Syntax**

```
typedef void (*FRTToolOnDeactivate)(  
    FS\_LPVOID clientData
```

);

**Description**

A callback for [FR\\_Tool](#). It is called when the tool no longer to be active tool.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return****Head file reference**

fr\_appExpT.h: 237

**Group**

[FR\\_ToolCallbacksRec](#)

**FRTToolOnKeyDown****Syntax**

```
typedef FS_BOOL (*FRTToolOnKeyDown)(  
    FS_LPVVOID clientData,  
    FS_UINT nKeyCode,  
    FS_UINT nFlags  
>;
```

**Description**

A callback for [FR\\_Tool](#). It is called when user presses a non-system key.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

nKeyCode	[In] Specifies the virtual-key code of the given key.
----------	---

---

nFlags	[In] Specifies the scan code, key-transition code, previous key state, and context code.
--------	--

---

**Return**

[TRUE](#) to halt further processing, or [FALSE](#) to continue.

**Head file reference**

fr\_appExpT.h: 252

**Group**

[FR\\_ToolCallbacksRec](#)

## FRTToolOnKeyUp

### Syntax

```
typedef FS_BOOL (*FRTToolOnKeyUp)(
    FS_LPVVOID clientData,
    FS_UINT nKeyCode,
    FS_UINT nFlags
);
```

### Description

A callback for [FR\\_Tool](#). It is called when user releases a non-system key.

### Parameter

clientData	[In] The user-supplied data.
nKeyCode	[In] Specifies the virtual-key code of the given key.
nFlags	[In] Specifies the scan code, key-transition code, previous key state, and context code.

### Return

[TRUE](#) to halt further processing, or [FALSE](#) to continue.

### Head file reference

fr\_appExpT.h: 267

### Group

[FR\\_ToolCallbacksRec](#)

## FRTToolOnChar

### Syntax

```
typedef FS_BOOL (*FRTToolOnChar)(
    FS_LPVVOID clientData,
    FS_UINT nChar,
    FS_UINT nFlags
);
```

### Description

A callback for [FR\\_Tool](#). It is called when a keystroke translates to a non-system character. This method is called before the [FRTToolOnKeyUp](#) method and after the [FRTToolOnKeyDown](#) method are called.

### Parameter

---

clientData	[In] The user-supplied data.
nChar	[In] Specifies the char code of the given key.
nFlags	[In] Specifies the scan code, key-transition code, previous key state, and context code.

---

**Return**

[TRUE](#) to halt further processing, or [FALSE](#) to continue.

**Head file reference**

fr\_appExpT.h: 283

**Group**

[FR\\_ToolCallbacksRec](#)

**FRTToolOnLeavePage****Syntax**

```
typedef void (*FRTToolOnLeavePage)(  
    FS\_LPVOID clientData,  
    FR\_PageView pageview  
>;
```

**Description**

A callback for [FR\\_Tool](#). It called when a tool leave a page view.

**Parameter**


---

clientData	[In] The user-supplied data.
pageview	[In] The page view.

---

**Return**

void

**Head file reference**

fr\_appExpT.h: 297

**Group**

[FR\\_ToolCallbacksRec](#)

**FRTToolIsEnabled****Syntax**

```
typedef FS_BOOL (*FRToolIsEnabled)(  
    FS_LPVVOID clientData  
);
```

**Description**

A callback for [FR\\_Tool](#) . Tests whether a tool is enable.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

[TRUE](#) means the tool is enable, otherwise not.

**Head file reference**

fr\_appExpT.h: 310

**Group**

[FR\\_ToolCallbacksRec](#)

**FRToolOnLButtonDown****Syntax**

```
typedef FS_BOOL (*FRToolOnLButtonDown)(  
    FS_LPVVOID clientData,  
    FR_PageView pageview,  
    FS_UINT nFlags,  
    FS_DevicePoint point  
);
```

**Description**

A callback for [FR\\_Tool](#) . It is called when user presses left mouse button.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

---

pageview	[In] The page view.
----------	---------------------

---

[In] Indicates whether various virtual keys are down. This parameter can be any combination of the following values:

nFlags

- -MK\_CONTROL Set if the *CTRL* key is down.
- -MK\_LBUTTON Set if the left mouse button is down.
- -MK\_MBUTTON Set if the middle mouse button is down.

- 
- -MK\_RBUTTON Set if the right mouse button is down.
  - -MK\_SHIFT Set if the *SHIFT* key is down.
- 

point [In] Specifies the x- and y-coordinate of the cursor. These coordinates are always relative to the upper-left corner of the window.

---

**Return**

[TRUE](#) to halt further processing, or [FALSE](#) to continue.

**Head file reference**

fr\_appExpT.h: 335

**Group**

[FR\\_ToolCallbacksRec](#)

**FRTToolOnLButtonUp****Syntax**

```
typedef FS_BOOL (*FRTToolOnLButtonUp)(
    FS_LPVVOID clientData,
    FR_PageView pageview,
    FS_UINT nFlags,
    FS_DevicePoint point
);
```

**Description**

A callback for [FR\\_Tool](#). It is called when user releases left mouse button.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

pageview	[In] The page view.
----------	---------------------

---

nFlags	[In] See <a href="#">FRTToolOnLButtonDown</a> () .
--------	--

---

point	[In] See <a href="#">FRTToolOnLButtonDown</a> () .
-------	--

---

**Return**

[TRUE](#) to halt further processing, or [FALSE](#) to continue.

**Head file reference**

fr\_appExpT.h: 351

**Group**  
[FR\\_ToolCallbacksRec](#)**FRTToolOnLButtonDblClk****Syntax**

```
typedef FS_BOOL (*FRTToolOnLButtonDblClk)(  
    FS\_LPVVOID clientData,  
    FR\_PageView pageview,  
    FS\_UINT nFlags,  
    FS\_DevicePoint point  
);
```

**Description**

A callback for [FR\\_Tool](#). It is called when user double-clicks the left mouse button.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

pageview	[In] The page view.
----------	---------------------

---

nFlags	[In] See <a href="#">FRTToolOnLButtonDown</a> ().
--------	---

---

point	[In] See <a href="#">FRTToolOnLButtonDown</a> ().
-------	---

---

**Return**

[TRUE](#) to halt further processing, or [FALSE](#) to continue.

**Head file reference**

fr\_appExpT.h: 367

**Group**[FR\\_ToolCallbacksRec](#)**FRTToolOnMouseMove****Syntax**

```
typedef FS_BOOL (*FRTToolOnMouseMove)(  
    FS\_LPVVOID clientData,  
    FR\_PageView pageview,  
    FS\_UINT nFlags,  
    FS\_DevicePoint point  
);
```

**Description**

A callback for [FR\\_Tool](#) . It is called when the mouse cursor or stylus moves.

**Parameter**


---

clientData	[In] The user-supplied data.
pageview	[In] The page view.
nFlags	[In] See <a href="#">FRToolOnLButtonDown</a> ().
point	[In] See <a href="#">FRToolOnLButtonDown</a> ().

---

**Return**

[TRUE](#) to halt further processing, or [FALSE](#) to continue.

**Head file reference**

fr\_appExpT.h: 383

**Group**

[FR\\_ToolCallbacksRec](#)

**FRToolOnRButtonDown****Syntax**

```
typedef FS_BOOL (*FRToolOnRButtonDown)(  
    FS_LPVVOID clientData,  
    FS_PageView pageview,  
    FS_UINT nFlags,  
    FS_DevicePoint point  
)
```

**Description**

A callback for [FR\\_Tool](#) . It is called when user presses right mouse button.

**Parameter**


---

clientData	[In] The user-supplied data.
pageview	[In] The page view.
nFlags	[In] See <a href="#">FRToolOnLButtonDown</a> ().
point	[In] See <a href="#">FRToolOnLButtonDown</a> ().

---

**Return**

[TRUE](#) to halt further processing, or [FALSE](#) to continue.

**Head file reference**

fr\_appExpT.h: 399

**Group**

[FR\\_ToolCallbacksRec](#)

**FRTToolOnRButtonUp****Syntax**

```
typedef FS_BOOL (*FRTToolOnRButtonUp)(  
    FS\_LPVVOID clientData,  
    FR\_PageView pageview,  
    FS\_UINT nFlags,  
    FS\_DevicePoint point  
)
```

**Description**

A callback for [FR\\_Tool](#). It is called when user releases right mouse button.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

pageview	[In] The page view.
----------	---------------------

---

nFlags	[In] See <a href="#">FRTToolOnLButtonDown</a> () .
--------	--

---

point	[In] See <a href="#">FRTToolOnLButtonDown</a> () .
-------	--

---

**Return**

[TRUE](#) to halt further processing, or [FALSE](#) to continue.

**Head file reference**

fr\_appExpT.h: 415

**Group**

[FR\\_ToolCallbacksRec](#)

**FRTToolOnRButtonDblClk****Syntax**

```
typedef FS_BOOL (*FRTToolOnRButtonDblClk)(  
    FS\_LPVVOID clientData,
```

```
FR_PageView pageview,  
FS_UINT nFlags,  
FS_DevicePoint point  
);
```

**Description**

A callback for [FR\\_Tool](#). It is called when user double-clicks the right mouse button.

**Parameter**

clientData	[In] The user-supplied data.
pageview	[In] The page view.
nFlags	[In] See <a href="#">FRToolOnLButtonDown</a> ().
point	[In] See <a href="#">FRToolOnLButtonDown</a> ().

**Return**

[TRUE](#) to halt further processing, or [FALSE](#) to continue.

**Head file reference**

fr\_appExpT.h: 431

**Group**

[FR\\_ToolCallbacksRec](#)

**FRTToolOnMouseWheel****Syntax**

```
typedef FS_BOOL (*FRToolOnMouseWheel)(  
    FS_LPVVOID clientData,  
    FR_PageView pageview,  
    FS_UINT nFlags,  
    FS_SHORT zDelta,  
    FS_DevicePoint point  
)
```

**Description**

A callback for [FR\\_Tool](#). It is called when a user rotates the mouse wheel and encounters the wheel's next notch.

**Parameter**

clientData	[In] The user-supplied data.
------------	------------------------------

---

pageview	[In] The page view.
nFlags	[In] See <a href="#">FRToolOnLButtonDown</a> () .
zDelta	[In] Indicates distance rotated.
point	[In] See <a href="#">FRToolOnLButtonDown</a> () .

---

**Return**

[TRUE](#) if mouse wheel scrolling is enabled; otherwise [FALSE](#) .

**Head file reference**

fr\_appExpT.h: 448

**Group**

[FR\\_ToolCallbacksRec](#)

**FRToolOnDraw****Syntax**

```
typedef FS_BOOL (*FRToolOnDraw)(  
    FS\_LPVOID clientData,  
    FR\_WinPort winPort,  
    FS\_DWORD dwFlags  
) ;
```

**Description**

A callback for [FR\\_Tool](#) . It's called when the page view is drew. Plug-in can draw its own content at this time.

**Parameter**


---

clientData	[In] The user-supplied data.
winPort	[In] The platform-depend things.
dwFlags	[In] The reserve flag, will be ignore.

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_appExpT.h: 463

**Group**

## [FR\\_ToolCallbacksRec](#)

### FRTToolIsProcessing

#### Syntax

```
typedef FS_BOOL (*FRTToolIsProcessing)(  
    FS_LPVVOID clientData  
)
```

#### Description

A callback for [FR\\_Tool](#). Tests whether a tool is processing.

#### Parameter

---

clientData	[In] The user-supplied.
------------	-------------------------

---

#### Return

[TRUE](#) means the tool is processing, otherwise not.

#### Head file reference

fr\_appExpT.h: 476

#### Group

## [FR\\_ToolCallbacksRec](#)

### FRTToolOnMouseHover

#### Syntax

```
typedef FS_BOOL (*FRTToolOnMouseHover)(  
    FS_LPVVOID clientData,  
    FR_PageView pageview,  
    FS_DevicePoint point  
)
```

#### Description

A callback for [FR\\_Tool](#). It is called when the cursor of the tool hovers over the client area of the window for the period of time.

#### Parameter

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

pageview	[In] The page view.
----------	---------------------

---

point	[In] See <a href="#">FRTToolOnLButtonDown</a> () .
-------	--

---

**Return**

[TRUE](#) if you process this event successfully, otherwise not.

**Head file reference**

fr\_appExpT.h: 491

**Group**

[FR\\_ToolCallbacksRec](#)

## FRToolIsWndCapturing

**Syntax**

```
typedef FS_BOOL (*FRToolIsWndCapturing)(  
    FS\_LPVVOID clientData  
)
```

**Description**

A callback for [FR\\_Tool](#). Determines whether continue to set the mouse capture to the PDF preview window after ButtonUp.

**Parameter**

---

clientData	[In] The user-supplied.
------------	-------------------------

---

**Return**

[TRUE](#) if you need continue to set the mouse capture to the PDF preview window after ButtonUp.

**Head file reference**

fr\_appExpT.h: 504

**Group**

[FR\\_ToolCallbacksRec](#)

## Functions

### Functions summary

**[FRToolGetName](#)**

Gets the name of specified tool.

**[FRToolNew](#)**

Creates a tool object.

**[FRToolRelease](#)**

Releases the specified tool.

**[FRToolSetAssociatedMousePtHandlerType](#)**

Sets the type of associated mouse point handler.

**[FRToolSetAssociatedSelectionHandlerType](#)**

---

Sets the type of associated selection handler.

## Functions detail

### FRTToolGetName

#### Syntax

```
void FRTToolGetName (
    FR\_Tool tool,
    FS\_BytString* outName
);
```

#### Description

Gets the name of specified tool.

#### Parameter

tool	[In] The tool object.
outName	[Out] The tool name buffer,filled by Reader.

#### Return

void

#### Head file reference

fr\_appTempl.h: 43

### FRTToolNew

#### Syntax

```
FR_Tool FRTToolNew (
    FS\_LPCSTR name,
    FR\_ToolCallbacks callbacks
);
```

#### Description

Creates a tool object.

#### Parameter

name	[In] The tool name. It may not be <a href="#">NULL</a> .
callbacks	[In] The event callbacks associate with a tool. When a event occurs, Reader will call a corresponding callback function.

#### Return

The new tool object or [NULL](#) if the tool with the *name* is exist.



**Head file reference**

fr\_appTempl.h: 21

**Related method**[FRToolRelease](#)[FRAppRegisterTool](#)

**Note:** If the tool name has been used, NULL will return.

**FRToolRelease****Syntax**

```
void FRToolRelease (
    FR\_Tool tool
);
```

**Description**

Releases the specified tool.

**Parameter**

---

tool	[In] The tool to release.
------	---------------------------

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 28

**FRToolSetAssociatedMousePtHandlerType****Syntax**

```
void FRToolSetAssociatedMousePtHandlerType (
    FR\_Tool tool,
    FS\_LPCSTR lpsType
);
```

**Description**

Sets the type of associated mouse point handler.

**Parameter**

---

tool	[In] The tool object.
------	-----------------------

---

---

lpsType	[Out] The type of associated mouse point handler.
---------	---

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 53

**Related method**[FRAppRegisterMousePtHandler](#)**FRTToolSetAssociatedSelectionHandlerType****Syntax**

```
void FRTToolSetAssociatedSelectionHandlerType (
    FR\_Tool tool,
    FS\_LPCSTR lpsType
);
```

**Description**

Sets the type of associated selection handler.

**Parameter**


---

tool	[In] The tool object.
------	-----------------------

---



---

lpsType	[Out] The type of associated selection handler.
---------	---

---

**Return**

void

**Head file reference**

fr\_appTempl.h: 64

**FR\_ToolBar**[Return from Used by](#)**Description**

[FR\\_ToolBar](#) is the Foxit Reader toolbar. A plug-in can also create fly-out toolbars that contain dditional buttons and attach the fly-out toolbars to existing button.

Plug-ins can add tool buttons to a toolbar or remove buttons form a toolbar. It can also show or hide a toolbar, and create new toolbars.

Buttons can be organized into groups which have same functions, with a separator between the groups. It is possible to implement a group in which only one button can be selected at a time. The logic of doing this is the plug-in's responsibility; the plug-in API

does not provide any means to automatically relate one button's state to another button's state.

A plug-in adds buttons to a toolbar by specifying the relative position of the button (before or after) to an existing button.

### Returned from

[FRAppGetToolBar](#)  
[FRAppGetToolBarByName](#)  
[FRToolBarNew](#)  
[FRToolBarNewFlyout](#)

### Used by

[FRToolBarSetFlyoutToolBar](#)  
[FRToolBarSetIcon](#)  
[FRToolBarAddButton](#)  
[FRToolBarCountButtons](#)  
[FRToolBarDock](#)  
[FRToolBarGetButton](#)  
[FRToolBarGetButtonByName](#)  
[FRToolBarGetName](#)  
[FRToolBarGetTitle](#)  
[FRToolBarHideButtonInBrowser](#)  
[FRToolBarHideToolBar](#)  
[FRToolBarHideToolBar](#)  
[FRToolBarInsertButton](#)  
[FRToolBarIsDisable](#)  
[FRToolBarIsFlyOutToolbar](#)  
[FRToolBarIsVisible](#)  
[FRToolBarRelease](#)  
[FRToolBarRemoveButton](#)  
[FRToolBarSetDefaultToolbar](#)  
[FRToolBarSetDisable](#)  
[FRToolBarSetTitle](#)  
[FRToolBarShowToolBar](#)  
[FRToolBarShowToolButton](#)  
[FRToolBarUpdateButtonStates](#)

## Definitions

### Definitions summary

#### [\*\*FR\\_TOOLBAR\\_NAME\\_ADVANCED\\_TOOLS\*\*](#)

Advanced Tools

#### [\*\*FR\\_TOOLBAR\\_NAME\\_BASIC\\_TOOLS\*\*](#)

Basic Tools

#### [\*\*FR\\_TOOLBAR\\_NAME\\_DIGITAL\\_TOOLS\*\*](#)

Digital Signature Tools

#### [\*\*FR\\_TOOLBAR\\_NAME\\_DRAWING2\\_TOOLS\*\*](#)

Drawing

#### [\*\*FR\\_TOOLBAR\\_NAME\\_DRAWING\\_TOOLS\*\*](#)

Drawing Markup Tools

**FR\_TOOLBAR\_NAME\_FATCH\_TOOLS**

FileAttachment Tools

**FR\_TOOLBAR\_NAME\_FAVORITE\_TOOLS**

Favorite Tools

**FR\_TOOLBAR\_NAME\_FILE\_TOOLS**

File

**FR\_TOOLBAR\_NAME\_FIND\_TOOLS**

Find ToolBar

**FR\_TOOLBAR\_NAME\_FLYOUTZOOM\_TOOLS**

Zoom Flyout

**FR\_TOOLBAR\_NAME\_FORM\_TOOLS**

Form Tools

**FR\_TOOLBAR\_NAME\_FORMAT\_TOOLS**

Format Tools

**FR\_TOOLBAR\_NAME\_FREETEXT\_TOOLS**

Typewriter Tools

**FR\_TOOLBAR\_NAME\_FULLSCREEN\_TOOLS**

Full Screen

**FR\_TOOLBAR\_NAME\_LINK\_TOOLS**

Link Tools

**FR\_TOOLBAR\_NAME\_MARKUP\_TOOLS**

Commenting Tools

**FR\_TOOLBAR\_NAME\_MARKUPS\_TOOS**

Markup Tools

**FR\_TOOLBAR\_NAME\_MEAS\_TOOLS**

Measure Tools

**FR\_TOOLBAR\_NAME\_MMEDIA\_TOOLS**

Multimedia Tools

**FR\_TOOLBAR\_NAME\_NAVIGATION\_TOOLS**

Navigation

**FR\_TOOLBAR\_NAME\_PROPERTY\_TOOLS**

Property Tools

**FR\_TOOLBAR\_NAME\_ROTATEVIEW\_TOOLS**

Rotate View

**FR\_TOOLBAR\_NAME\_SECURITY\_TOOLS**

Security Editing Tools

**FR\_TOOLBAR\_NAME\_STAMP\_TOOLS**

Stamp Tools

**FR\_TOOLBAR\_NAME\_TEXT\_TOOLS**

Smart Text Tools

**FR\_TOOLBAR\_NAME\_TEXTVIEWER\_TOOLS**

Text Viwer

**FR\_TOOLBAR\_NAME\_ZOOM\_TOOLS**

Zoom

Definitions detail

**FR\_TOOLBAR\_NAME\_ADVANCED\_TOOLS**

**Syntax**

```
#define FR_TOOLBAR_NAME_ADVANCED_TOOLS "Advanced Editing"
```

**Description**

Advanced Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 158

## FR\_TOOLBAR\_NAME\_BASIC\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_BASIC_TOOLS "Basic Tools"
```

**Description**

Basic Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 110

## FR\_TOOLBAR\_NAME\_DIGITAL\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_DIGITAL_TOOLS "Digital Signature Tools"
```

**Description**

Digital Signature Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 189

## FR\_TOOLBAR\_NAME\_DRAWING2\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_DRAWING2_TOOLS "Drawing"
```

**Description**

Drawing

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 168

## FR\_TOOLBAR\_NAME\_DRAWING\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_DRAWING_TOOLS "Drawing Markup Tools"
```

**Description**

Drawing Markup Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 143

## FR\_TOOLBAR\_NAME\_FATCH\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_FATCH_TOOLS "FileAttachment Tools"
```

**Description**

FileAttachment Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 162

## FR\_TOOLBAR\_NAME\_FAVORITE\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_FAVORITE_TOOLS "Favorite Tools"
```

**Description**

Favorite Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 134

## FR\_TOOLBAR\_NAME\_FILE\_TOOLS

### Syntax

```
#define FR_TOOLBAR_NAME_FILE_TOOLS "File"
```

### Description

File

### Group

[FRToolbarNames](#)

### Head file reference

fr\_barExpT.h: 113

## FR\_TOOLBAR\_NAME\_FIND\_TOOLS

### Syntax

```
#define FR_TOOLBAR_NAME_FIND_TOOLS "FindToolBar"
```

### Description

FindToolBar

### Group

[FRToolbarNames](#)

### Head file reference

fr\_barExpT.h: 180

## FR\_TOOLBAR\_NAME\_FLYOUTZOOM\_TOOLS

### Syntax

```
#define FR_TOOLBAR_NAME_FLYOUTZOOM_TOOLS "Zoom Flyout"
```

### Description

Zoom Flyout

### Group

[FRToolbarNames](#)

### Head file reference

fr\_barExpT.h: 131

## FR\_TOOLBAR\_NAME\_FORM\_TOOLS

### Syntax

```
#define FR_TOOLBAR_NAME_FORM_TOOLS "Form Tools"
```

**Description**

Form Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 140

## FR\_TOOLBAR\_NAME\_FORMAT\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_FORMAT_TOOLS "Format Tools"
```

**Description**

Format Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 174

## FR\_TOOLBAR\_NAME\_FREETEXT\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_FREETEXT_TOOLS "Typewriter Tools"
```

**Description**

Typewriter Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 177

## FR\_TOOLBAR\_NAME\_FULLSCREEN\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_FULLSCREEN_TOOLS "Full Screen"
```

**Description**

Full Screen

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 125

## FR\_TOOLBAR\_NAME\_LINK\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_LINK_TOOLS "Link Tools"
```

**Description**

Link Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 149

## FR\_TOOLBAR\_NAME\_MARKUP\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_MARKUP_TOOLS "Commenting Tools"
```

**Description**

Commenting Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 137

## FR\_TOOLBAR\_NAME\_MARKUPS\_TOOS

**Syntax**

```
#define FR_TOOLBAR_NAME_MARKUPS_TOOS "Markup Tools"
```

**Description**

Markup Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 165

## FR\_TOOLBAR\_NAME\_MEAS\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_MEAS_TOOLS "Measure Tools"
```

**Description**

Measure Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 152

## FR\_TOOLBAR\_NAME\_MMEDIA\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_MMEDIA_TOOLS "Multimedia Tools"
```

**Description**

Multimedia Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 155

## FR\_TOOLBAR\_NAME\_NAVIGATION\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_NAVIGATION_TOOLS "Navigation"
```

**Description**

Navigation

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 116

## FR\_TOOLBAR\_NAME\_PROPERTY\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_PROPERTY_TOOLS "Property Tools"
```

**Description**

Property Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 186

## FR\_TOOLBAR\_NAME\_ROTATEVIEW\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_ROTATEVIEW_TOOLS "Rotate View"
```

**Description**

Rotate View

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 122

## FR\_TOOLBAR\_NAME\_SECURITY\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_SECURITY_TOOLS "Security Editing Tools"
```

**Description**

Security Editing Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 183

## FR\_TOOLBAR\_NAME\_STAMP\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_STAMP_TOOLS "Stamp Tools"
```

**Description**

Stamp Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 171

## FR\_TOOLBAR\_NAME\_TEXT\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_TEXT_TOOLS "Smart Text Tools"
```

**Description**

Smart Text Tools

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 146

## FR\_TOOLBAR\_NAME\_TEXTVIEWER\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_TEXTVIEWER_TOOLS "Text Viwer"
```

**Description**

Text Viwer

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 128

## FR\_TOOLBAR\_NAME\_ZOOM\_TOOLS

**Syntax**

```
#define FR_TOOLBAR_NAME_ZOOM_TOOLS "Zoom"
```

**Description**

Zoom

**Group**

[FRToolbarNames](#)

**Head file reference**

fr\_barExpT.h: 119

## Functions

### Functions summary

#### [FRToolBarAddButton](#)

Adds a button to the end of the toolbar. Using [FRToolBarAddButton \(\)](#) to insert a button into the toolbar.

#### [FRToolBarCountButtons](#)

Counts the button number of the a toolbar.

#### [FRToolBarDock](#)

Docks the toolbar.

#### [FRToolBarGetButton](#)

Gets the specified button.

#### [FRToolBarGetButtonByName](#)

Gets the toolbar button that has the specified name.

#### [FRToolBarGetName](#)

Gets the toolbar's name.

#### [FRToolBarGetTitle](#)

Gets the title of the specified toolbar.

#### [FRToolBarHideButtonInBrowser](#)

Hides the specified button in browser mode or not.

#### [FRToolBarHideToolBar](#)

Hides a toolbar.

#### [FRToolBarHideToolBarButton](#)

Hides a existing toolbar button.

#### [FRToolBarInsertButton](#)

Inserts a button into a toolbar.

#### [FRToolBarIsDisable](#)

Checks whether the toolbar is disabled or not.

#### [FRToolBarIsFlyOutToolbar](#)

Whether the toolbar is a fly-out toolbar or not.

#### [FRToolBarIsVisible](#)

Checks whether the toolbar is visible or not.

#### [FRToolBarNew](#)

Creates a new named toolbar, or [NULL](#) if a toolbar with the *name* exists.

#### [FRToolBarNewFlyout](#)

Creates a new named flyout toolbar, or [NULL](#) if a toolbar with the *name* exists. A fly-out toolbar is used to append to a toolbar's button with the drop-down style.

#### [FRToolBarRelease](#)

Removes a specified toolbar and releases it.

#### [FRToolBarRemoveButton](#)

Removes the specified button from the toolbar, but does not destroy the button. Call [FRToolBarUpdateButtonStates \(\)](#) after removing a button to update the toolbar.

#### [FRToolBarSetDefaultToolbar](#)

Sets the toolbar as a default toolbar that will be shown when application starts up.

#### [FRToolBarSetDisable](#)

Disables the toolbar.

#### [FRToolBarSetMenuTitle](#)

Sets the menu title.

#### [FRToolBarSetTitle](#)

Sets the title of specified toolbar.

**[FRToolBarShowToolBar](#)**

Shows a toolbar.

**[FRToolBarShowToolBar](#)**

Shows a existing toolbar button.

**[FRToolBarUpdateButtonStates](#)**

Updates the states of buttons when the toolbar is modified.

## Functions detail

### [FRToolBarAddButton](#)

**Syntax**

```
FS_BOOL FRToolBarAddButton (
    FRToolBar toolbar,
    FRToolButton btn
);
```

**Description**

Adds a button to the end of the toolbar. Using [FRToolBarAddButton \(\)](#) to insert a button into the toolbar.

**Parameter**

---

toolbar	[In] The toolbar into which a button is added.
---------	--

---

btn	[In] The button to add to the toolbar.
-----	--

---

**Return**

[TRUE](#) if success, otherwise not.

**Head file reference**

fr\_barTempl.h: 368

**Related method**

[FRToolBarInsertButton](#)

[FRToolBarRemoveButton](#)

### [FRToolBarCountButtons](#)

**Syntax**

```
FS_INT32 FRToolBarCountButtons (
    FRToolBar toolbar
);
```

**Description**

Counts the button number of the a toolbar.

**Parameter**

---

toolbar	[In] The toolbar whose button count is obtained.
---------	--

---

**Return**

The button number of the specified toolbar.

**Head file reference**

fr\_barTempl.h: 267

**Related method**

[FRToolBarGetButton](#)

**FRTToolBarDock****Syntax**

```
void FRTToolBarDock (
    FRToolBar toolbar
);
```

**Description**

Docks the toolbar.

**Parameter**

---

toolbar	[In] The input toolbar.
---------	-------------------------

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 529

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRTToolBarGetButton****Syntax**

```
FR_ToolButton FRTToolBarGetButton (
    FRToolBar toolbar,
    FS\_INT32 index
);
```

**Description**

Gets the specified button.

**Parameter**

---

toolbar	[In] The toolbar whose button is obtained.
index	[In] The index of the button. The index range is 0 to ( <a href="#">FRToolBarCountButton</a> ()-1).

---

**Return**

The specified button.

**Head file reference**

fr\_barTempl.h: 58

**Related method**

[FRToolBarCountButtons](#)

[FRToolBarGetButtonByName](#)

**FRToolBarGetButtonByName****Syntax**

```
FR_ToolButton FRToolBarGetButtonByName (
    FRToolBar toolbar,
    const FS\_CHAR* name
);
```

**Description**

Gets the toolbar button that has the specified name.

**Parameter**


---

toolbar	[In] The input toolbar.
name	[In] The name for the button to get.

---

**Return**

The button with the specified name, if the name is not found, the return value is [NULL](#) .

**Head file reference**

fr\_barTempl.h: 58

**FRToolBarGetName****Syntax**

```
FS_BOOL FRToolBarGetName (
    FRToolBar toolbar,
    FS\_ByteString* outName
);
```

**Description**

Gets the toolbar's name.

**Parameter**


---

toolbar	[In] The toolbar whose name is obtained.
outName	[Out] (Filled by this method) The string buffer to receive the toolbar's name.

---

**Return**

[TRUE](#) if the *outName* is filled successfully, otherwise not.

**Head file reference**

fr\_barTempl.h: 301

**Related method**

[FRAppGetToolBarByName](#)

**FRToolBarSetTitle****Syntax**

```
FS_BOOL FRToolBarSetTitle (
    FRToolBar toolbar,
    FS\_WideString* outTitle
);
```

**Description**

Gets the title of the specified toolbar.

**Parameter**


---

toolbar	[In] The toolbar whose title is obtained.
outTitle	[Out] (Filled by this method) A wide string buffer to receive the toolbar's title.

---

**Return**

[TRUE](#) if the *outTitle* is filled successfully, otherwise not.

**Head file reference**

fr\_barTempl.h: 335

**Related method**

[FRToolBarSetTitle](#)

## FRTToolBarHideButtonInBrowser

### Syntax

```
FS_BOOL FRTToolBarHideButtonInBrowser (
    FRToolBar toolbar,
    FS_LPCSTR csName,
    FS_BOOL bHide
);
```

### Description

Hides the specified button in browser mode or not.

### Parameter

toolbar	[In] The input toolbar.
csName	[In] The specified button name.
bHide	[In] Sets it TRUE if you want to hide the button in browser mode, otherwise FALSE.

### Return

[TRUE](#) for success, otherwise failure.

### Head file reference

fr\_barTempl.h: 487

### Since

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRTToolBarHideToolBar

### Syntax

```
void FRTToolBarHideToolBar (
    FRToolBar toolbar
);
```

### Description

Hides a toolbar.

### Parameter

toolbar	[In] The toolbar to hide.
---------	---------------------------

### Return

void

**Head file reference**

fr\_barTempl.h: 404

**Related method**[FRToolBarShowToolBar](#)**FRToolBarHideToolBarButton****Syntax**

```
void FRToolBarHideToolBarButton (
    FRToolBar toolbar,
    FRToolBarButton btn
);
```

**Description**

Hides a existing toolbar button.

**Parameter**

---

toolbar	[In] The toolbar whose button is hided.
---------	---

---

btn	[In] The button to hide.
-----	--------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 420

**Related method**[FRToolBarShowToolBarButton](#)**FRToolBarInsertButton****Syntax**

```
FS_BOOL FRToolBarInsertButton (
    FRToolBar toolbar,
    FRToolBarButton btn,
    FS\_INT32 iInsertAt
);
```

**Description**

Inserts a button into a toolbar.

**Parameter**

---

toolbar	[In] The toolbar into which a button is inserted.
---------	---

---

btn	[In] The button to insert into the toolbar.
-----	---

---

iInsertAt	[In] The specified index for inserting.
-----------	---

---

**Return**

[TRUE](#) if success, otherwise [FALSE](#).

**Head file reference**

fr\_barTempl.h: 375

**Related method**

[FTToolBarAddButton](#)

[FRToolBarRemoveButton](#)

**FRToolBarIsDisable****Syntax**

```
FS_BOOL FRToolBarIsDisable (
    FRToolBar toolbar
);
```

**Description**

Checks whether the toolbar is disabled or not.

**Parameter**


---

toolbar	[In] The input toolbar.
---------	-------------------------

---

**Return**

[TRUE](#) if the toolbar is disabled, otherwise not.

**Head file reference**

fr\_barTempl.h: 509

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRToolBarIsFlyOutToolbar****Syntax**

```
FS_BOOL FRToolBarIsFlyOutToolbar (
    FRToolBar toolbar
);
```

**Description**

Whether the toolbar is a fly-out toolbar or not.

**Parameter**

---

toolbar	[In] The input toolbar.
---------	-------------------------

---

**Return**

TRUE means that the toolbar is a fly-out toolbar, otherwise not.

**Head file reference**

fr\_barTempl.h: 459

**FRTToolBarIsVisible****Syntax**

```
FS_BOOL FRTToolBarIsVisible (
    FRToolBar toolbar
);
```

**Description**

Checks whether the toolbar is visible or not.

**Parameter**

---

toolbar	[In] The input toolbar.
---------	-------------------------

---

**Return**

TRUE if the toolbar is visible, otherwise not.

**Head file reference**

fr\_barTempl.h: 499

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRTToolBarNew****Syntax**

```
FRToolBar FRTToolBarNew (
    const FS_CHAR* name,
    FS_LPCWSTR title,
    void* pParentWnd
);
```

**Description**

Creates a new named toolbar, or [NULL](#) if a toolbar with the *name* exists.

#### Parameter

---

name	[In] The name of the toolbar. It may not be <a href="#">NULL</a> .
------	--

---

title	[In] The title of the toolbar. It may not be <a href="#">NULL</a> .
-------	---

---

pParentWnd	[In] A pointer to the parent window. It represents the <i>MFC CWnd*</i> . Sets it to NULL as default. You can get the parent window through <a href="#">FRDocGetUIParentWnd</a> .
------------	---

---

#### Return

The new [FRToolBar](#) object.

#### Head file reference

fr\_barTempl.h: 219

#### Related method

[FRToolBarRelease](#)

[FRToolBarNew](#)

[FRToolBarCountButtons](#)

## FRTToolBarNewFlyout

#### Syntax

```
FRToolBar FRTToolBarNewFlyout (
    const FS_CHAR* name,
    FS_LPCWSTR title,
    void* pParentWnd
);
```

#### Description

Creates a new named flyout toolbar, or [NULL](#) if a toolbar with the *name* exists. A fly-out toolbar is used to append to a toolbar's button with the drop-down style.

#### Parameter

---

name	[In] The name of the flyout toolbar. It may not be <a href="#">NULL</a> .
------	---

---

title	[In] The title of the flyout toolbar. It may not be <a href="#">NULL</a> .
-------	--

---

pParentWnd	[In] A pointer to the parent window. It represents the <i>MFC CWnd*</i> . Sets it to NULL as default. You can get the parent window through <a href="#">FRDocGetUIParentWnd</a> .
------------	---

---

**Return**

The newly created fly-out toolbar.

**Head file reference**

fr\_barTempl.h: 219

**Related method**

[FRToolBarRelease](#)

[FRToolBarNew](#)

[FRToolBarCountButtons](#)

[FRToolBarSetFlyoutToolBar](#)

**FRToolBarRelease****Syntax**

```
void FRToolBarRelease (
    FRToolBar toolbar,
    void* pParentWnd
);
```

**Description**

Removes a specified toolbar and releases it.

**Parameter**

---

toolbar	[In] The toolbar to be released.
---------	----------------------------------

---

pParentWnd	[In] A pointer to the parent window that you must specify. It represents the <i>MFC CWnd*</i> . You can get the parent window through <a href="#">FRDocGetUIParentWnd</a> .
------------	---

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 265

**Related method**

[FRToolBarNew](#)

**FRToolBarRemoveButton****Syntax**

```
FS_BOOL FRToolBarRemoveButton (
    FRToolBar toolbar,
    FRToolBar btn
);
```

**Description**

Removes the specified button from the toolbar, but does not destroy the button. Call [FRToolBarUpdateButtonStates](#) () after removing a button to update the toolbar.

**Parameter**

---

toolbar	[In] The toolbar whose button is removed.
---------	---

---

## btn

[In] The button to remove.

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_barTempl.h: 376

**Related method**

[FRToolBarUpdateButtonStates](#)  
[FRToolBarAddButton](#)  
[FRToolBarInsertButton](#)

**FRToolBarSetDefaultToolbar****Syntax**

```
void FRToolBarSetDefaultToolbar (
    FRToolBar toolbar
);
```

**Description**

Sets the toolbar as a default toolbar that will be shown when application starts up.

**Parameter**

---

toolbar	[In] The input toolbar.
---------	-------------------------

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 478

**FRToolBarSetDisable****Syntax**

```
void FRToolBarSetDisable (
    FRToolBar toolbar
```

);

**Description**

Disables the toolbar.

**Parameter**

---

toolbar	[In] The input toolbar.
---------	-------------------------

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 519

**Since**

[SDK LATEEST VERSION > 2.0](#)

**FRTToolBarSetTitle****Syntax**

```
void FRTToolBarSetTitle (
    FRToolBar toolbar,
    FS\_LPCWSTR menutitle
);
```

**Description**

Sets the menu title.

**Parameter**

---

toolbar	[In] The input toolbar.
---------	-------------------------

---

---

menutitle	[In] The input menu title.
-----------	----------------------------

---

**Return**

void.

**Head file reference**

fr\_barTempl.h: 468

**FRTToolBarSetTitle****Syntax**

```
void FRTToolBarSetTitle (
    FRToolBar toolbar,
```

```
FS_LPCWSTR title  
);
```

**Description**

Sets the title of specified toolbar.

**Parameter**

---

toolbar	[In] The toolbar whose title is set.
---------	--------------------------------------

---

title	[In] The title to set.
-------	------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 341

**Related method**

[FRToolBarSetTitle](#)

**FRToolBarShowToolBar****Syntax**

```
void FRToolBarShowToolBar (  
    FRToolBar toolbar  
);
```

**Description**

Shows a toolbar.

**Parameter**

---

toolbar	[In] The toolbar to show.
---------	---------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 394

**Related method**

[FRToolBarHideToolBar](#)

**FRToolBarShowToolButton**

**Syntax**

```
void FRToolBarShowToolBar (  
    FRToolBar toolbar,  
    FRToolButton btn  
) ;
```

**Description**

Shows a existing toolbar button.

**Parameter**

---

toolbar	[In] The toolbar whose button is showed.
---------	--

---

btn	[In] The button to show.
-----	--------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 414

**Related method**

[FRToolBarHideToolButton](#)

**FRToolBarUpdateButtonStates****Syntax**

```
void FRToolBarUpdateButtonStates (  
    FRToolBar toolbar  
) ;
```

**Description**

Updates the states of buttons when the toolbar is modified.

**Parameter**

---

toolbar	[In] The input toolbar.
---------	-------------------------

---

**Return**

void

**Head file reference**

fr\_barTempl.h: 438

## FR\_ToolButton

## [Return from Used by](#)

### Description

An [FR\\_ToolButton](#) is a button in the Foxit Reader's toolbar. Like menu items, the procedure that executes when the button is clicked can be set by a plug-in. Although not required, there generally is a menu item corresponding to each button, allowing users to select a function using either the button or the menu item.

A plug-in can invoke a button as if a user clicked it. Buttons can be enabled (selectable) or disabled (grayed out), and can be marked (selected). Each button also has an icon that appears in the toolbar. [FR\\_ToolButton](#) objects frequently, but not always, change the active tool. For example, the button that selects the snapshot tool changes the active tool.

### Returned from

[FRToolBarGetButton](#)  
[FRToolBarGetButtonByName](#)  
[FRToolButtonNew](#)

### Used by

[FRToolBarAddButton](#)  
[FRToolBarHideToolBar](#)  
[FRToolBarInsertButton](#)  
[FRToolBarRemoveButton](#)  
[FRToolBarShowToolBar](#)  
[FRToolButtonExecuteProc](#)  
[FRToolButtonGetClientData](#)  
[FRToolButtonGetLabelText](#)  
[FRToolButtonGetMapId](#)  
[FRToolButtonGetName](#)  
[FRToolButtonIsSeparator](#)  
[FRToolButtonIsVisible](#)  
[FRToolButtonRelease](#)  
[FRToolButtonSetClientData](#)  
[FRToolButtonSetDropDownProc](#)  
[FRToolButtonSetEnableProc](#)  
[FRToolButtonSetExecuteProc](#)  
[FRToolButtonSetFlyoutToolBar](#)  
[FRToolButtonSetHelpText](#)  
[FRToolButtonSetIcon](#)  
[FRToolButtonSetLabelText](#)  
[FRToolButtonSetMarkedProc](#)

### Callbacks

#### Callbacks summary

##### [FRBtnDropDownProc](#)

The callback is invoked by Foxit Reader when user clicks the drop-down arrow displayed next to the button image or text.

#### Callbacks detail

## FRBtnDropDownProc

### Syntax

```
typedef FS_BOOL (*FRBtnDropDownProc)(
    void clientData,
    FS_Rect rect
);
```

### Description

The callback is invoked by Foxit Reader when user clicks the drop-down arrow displayed next to the button image or text.

### Parameter

---

clientData	[In] The user-supplied data. Plug-ins can set the data by invoking <a href="#">FRToolButtonSetClientData</a> ();
------------	--

---

rect	[In] The rectangle of the drop-down button.
------	---

---

### Return

[TRUE](#) means it is handled successfully, otherwise not.

### Head file reference

fr\_barExpT.h: 76

## Functions

### Functions summary

#### [FRToolButtonExecuteProc](#)

Executes the [FRExecuteProc](#) () associated with button. It does nothing if [FRBtnEnableProc](#) () returns [FALSE](#) .

#### [FRToolButtonGetClientData](#)

Gets the user-supplied data structure set to tool button using [FRToolButtonSetClientData](#) ().

#### [FRToolButtonGetLabelText](#)

Gets the label text of specified button.

#### [FRToolButtonGetMapId](#)

Gets the map id of the button.

#### [FRToolButtonGetName](#)

Gets the name of the specified button.

#### [FRToolButtonIsSeparator](#)

Tests whether a toolbar button is a separator or a normal button.

#### [FRToolButtonIsVisible](#)

Tests whether a toolbar button is visible.

#### [FRToolButtonNew](#)

Creates a toolbar button or a separator with the specified name.

#### [FRToolButtonRelease](#)

Removes a specified toolbar button and releases it. Call [FRToolButtonUpdateButtonStates\(\)](#) after removing a button to update the toolbar.

#### [\*\*FRToolButtonSetClientData\*\*](#)

Sets the user-supplied data for each user-supplied procedure.

#### [\*\*FRToolButtonSetDropDownProc\*\*](#)

Sets a [FRBtnDropDownProc\(\)](#) associated with a button.

#### [\*\*FRToolButtonSetEnableProc\*\*](#)

Sets a [FRBtnEnableProc\(\)](#) associated with a button. This routine determines whether a toolbar button can be selected.

#### [\*\*FRToolButtonSetExecuteProc\*\*](#)

Sets the user-supplied procedure to call to perform the button's intended function.

#### [\*\*FRToolButtonSetFlyoutToolBar\*\*](#)

Sets the fly-out toolbar to a specified button.

#### [\*\*FRToolButtonSetHelpText\*\*](#)

Sets the help text of the specified button.

#### [\*\*FRToolButtonSetIcon\*\*](#)

Sets a new bitmap for a toolbar button. The size of bitmap which to be set to a button is different between common toolbar mode and tabbed toolbar mode, with common mode, the size of bitmap is 24 \* 24, but 32 \* 32 with tabbed mode.

#### [\*\*FRToolButtonSetLabelText\*\*](#)

Sets the label text of the specified button.

#### [\*\*FRToolButtonSetMarkedProc\*\*](#)

Sets a [FRBtnCheckProc\(\)](#) associated with a button.

## Functions detail

### [FRToolButtonExecuteProc](#)

#### **Syntax**

```
void FRToolButtonExecuteProc (
    FR\_ToolButton btn
);
```

#### **Description**

Executes the [FRExecuteProc\(\)](#) associated with button. It does nothing if [FRBtnEnableProc\(\)](#) returns [FALSE](#).

#### **Parameter**

---

btn	[In] The button whose execute proc is executed.
-----	---

---

#### **Return**

void

#### **Head file reference**

fr\_barTempl.h: 202

#### **Related method**

[FRToolButtonSetExecuteProc](#)

## FRTadioButtonGetClientData

### Syntax

```
void* FRTadioButtonGetClientData (
    FR\_ToolButton btn
);
```

### Description

Gets the user-supplied data structure set to tool button using [FRTadioButtonSetClientData](#) ().

### Parameter

---

btn	[In] The button whose client-data is got.
-----	---

---

### Return

A pointer to a user-supplied data structure. It returns [NULL](#) if no client data to be set.

### Head file reference

fr\_barTempl.h: 191

### Related method

[FRTadioButtonSetClientData](#)

## FRTadioButtonGetLabelText

### Syntax

```
FS_BOOL FRTadioButtonGetLabelText (
    FR\_ToolButton btn,
    FS\_WideString* outLabelText
);
```

### Description

Gets the label text of specified button.

### Parameter

---

btn	[In] The button whose label is obtained.
-----	--

---

outLabelText	[Out] (Filled by the method) A Unicode string buffer to receive the button's label text.
--------------	--

---

### Return

[TRUE](#) if the *outLabelText* is filled successfully, otherwise not.

### Head file reference

fr\_barTempl.h: 63

**Related method**

[FRToolButtonSetLableText](#)

**FRToolButtonGetMapId****Syntax**

```
FS_DWORD FRToolButtonGetMapId (
    FR\_ToolButton btn
);
```

**Description**

Gets the map id of the button.

**Parameter**

---

btn	[In] The input button.
-----	------------------------

---

**Return**

The map id of the button.

**Head file reference**

fr\_barTempl.h: 224

**Related method**

[FRWndProviderOnCmdMsg](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.0](#)

**FRToolButtonGetName****Syntax**

```
FS_BOOL FRToolButtonGetName (
    FR\_ToolButton btn,
    FS\_ByteString* outName
);
```

**Description**

Gets the name of the specified button.

**Parameter**

---

btn	[In] The button whose name is obtained.
-----	---

---

outName	[Out] (Filled by the method) A string buffer to receive the name.
---------	---

---

**Return**

[TRUE](#) if the *outName* is filled successfully, otherwise not.

**Head file reference**

fr\_barTempl.h: 52

**Related method**

[FRToolBarGetButtonByName](#)

**FRToolBarIsSeparator****Syntax**

```
FS_BOOL FRToolBarIsSeparator (
    FR\_ToolButton btn
);
```

**Description**

Tests whether a toolbar button is a separator or a normal button.

**Parameter**

---

btn	[In] The button to test.
-----	--------------------------

---

**Return**

[TRUE](#) if a button is a separator, otherwise not.

**Head file reference**

fr\_barTempl.h: 112

**FRToolBarIsVisible****Syntax**

```
FS_BOOL FRToolBarIsVisible (
    FR\_ToolButton btn
);
```

**Description**

Tests whether a toolbar button is visible.

**Parameter**

---

btn	[In] The button to test.
-----	--------------------------

---

**Return**

[TRUE](#) if a button is visible, otherwise not.

**Head file reference**

fr\_barTempl.h: 121

**FRToolButtonNew****Syntax**

```
FR_ToolButton FRToolButtonNew (
    const FS_CHAR* name,
    FS_BOOL bSeparate,
    void* pParentWnd
);
```

**Description**

Creates a toolbar button or a separator with the specified name.

**Parameter**

name	[In] The button's name.
bSeparate	[In] A flag indicate whether a button or a separator is to be created.
pParentWnd	[In] A pointer to the parent window. It represents the <i>MFC CWnd*</i> . Sets it to NULL as default. You can get the parent window through <a href="#">FRDocGetUIParentWnd</a> . If <a href="#">TRUE</a> , the new button is a separator used to leave space between groups of related buttons. If <a href="#">FALSE</a> , the button is a normal button.

**Return**

The newly toolbar button.

**Head file reference**

fr\_barTempl.h: 21

**Related method**

[FRToolButtonRelease](#)

[FRToolButtonSetExcuteProc](#)

[FRToolButtonSetEnableProc](#)

[FRToolButtonSetMarkedProc](#)

**FRToolButtonRelease****Syntax**

```
void FRToolButtonRelease (
```

```
FR_ToolButton btn,  
void* pParentWnd  
);
```

### Description

Removes a specified toolbar button and releases it. Call [FRToolButtonUpdateButtonStates\(\)](#) after removing a button to update the toolbar.

### Parameter

btn	[In] The button to release.
pParentWnd	[In] A pointer to the parent window that you must specify. It represents the <i>MFC CWnd*</i> . You can get the parent window through <a href="#">FRDocGetUIParentWnd</a> .

### Return

void

### Head file reference

fr\_barTempl.h: 31

### Related method

[FRToolButtonNew](#)

## FRToolButtonSetClientData

### Syntax

```
FS_BOOL FRToolButtonSetClientData (  
    FR_ToolButton btn,  
    void* clientData,  
    FRFreeDataProc callback  
)
```

### Description

Sets the user-supplied data for each user-supplied procedure.

### Parameter

btn	[In] The button whose client-data is set.
clientData	[In] A pointer to user-supplied data to pass to <a href="#">FRExecuteProc()</a> , <a href="#">FRBtnEnableProc()</a> , <a href="#">FRBtnCheckProc()</a> , or <a href="#">FRBtnDropDownProc()</a> . The data type may be a class or a struct that contain each client data to pass to each user-supplied procedure.

---

callback	[In] It will be called when the tool button is to be destroyed.
----------	---

---

**Return**

TRUE means successful, otherwise not.

**Head file reference**

fr\_barTempl.h: 140

**Since**

SDK\_LATEEST\_VERSION > 2.0

**FRToolButtonSetDropDownProc****Syntax**

```
FS_BOOL FRToolButtonSetDropDownProc (
    FR_ToolButton btn,
    FRBtnDropDownProc proc
);
```

**Description**

Sets a FRBtnDropDownProc () associated with a button.

**Parameter**

---

btn	[In] The button whose <u>FRBtnDropDownProc</u> () is set.
-----	---

---

proc	[In] A user-supplied procedure to call when user click the drop-down arrow displayed next to the button image or text.
------	--

---

**Return**

TRUE means successful, otherwise not.

**Head file reference**

fr\_barTempl.h: 178

**Related method**

FRToolButtonSetClientData

**Note:** The user-supplied data passed to FRBtnDropDownProc() must be set if the callback FRBtnDropDownProc() need. Using FRToolButtonSetClientDate() to set it.

**FRToolButtonSetEnableProc****Syntax**

```
FS_BOOL FRToolButtonSetEnableProc (
    FR_ToolButton btn,
    FRComputeEnabledProc proc
```

```
 );
```

**Description**

Sets a [FRBtnEnableProc](#) () associated with a button. This routine determines whether a toolbar button can be selected.

**Parameter**


---

btn	[In] The button whose <a href="#">FRBtnEnableProc</a> () is set.
proc	[In] A user-supplied procedure to call whenever Reader needs to know whether a button should be enabled.

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_barImpl.h: 33

**Related method**

[FRToolButtonNew](#)

[FRToolButtonSetExecuteProc](#)

[FRToolButtonSetMarkedProc](#)

[FRToolButtonSetClientData](#)

**Note:** The user-supplied data to pass to FRBtnEnableProc() must be set if the callback FRBtnEnableProc() need. Using FRToolButtonSetClientDate() to set it.

**FRToolButtonSetExecuteProc****Syntax**

```
FS_BOOL FRToolButtonSetExecuteProc (
    FR\_ToolButton btn,
    FRExecuteProc proc
);
```

**Description**

Sets the user-supplied procedure to call to perform the button's intended function.

**Parameter**


---

btn	[In] The button whose intended function is set.
proc	[In] The user-supplied procedure to call when <i>btn</i> is clicked.

---

**Return**

[TRUE](#) means successful, otherwise not.

#### Head file reference

fr\_barTempl.h: 130

#### Related method

[FRToolButtonNew](#)

[FRToolButtonSetEnableProc](#)

[FRToolButtonSetMarkedProc](#)

[FRToolButtonSetClientData](#)

**Note:** The user-supplied data to pass to FRExecuteProc() must be set if the callback FRExecuteProc() needs. Using FRToolButtonSetClientDate() to set it.

### FRToolButtonSetFlyoutToolBar

#### Syntax

```
FS_BOOL FRToolButtonSetFlyoutToolBar (
    FR\_ToolButton btn,
    FRToolBar flyout
);
```

#### Description

Sets the fly-out toolbar to a specified button.

#### Parameter

btn	[In] The button to which the flyout toolbar is appended.
-----	--

flyout	[In] The flyout toolbar which to be appended to the <i>btn</i> .
--------	--

#### Return

[TRUE](#) means successful, otherwise not.

#### Head file reference

fr\_barTempl.h: 213

#### Related method

[FRToolBarNewFlyout](#)

### FRToolButtonSetHelpText

#### Syntax

```
FS_BOOL FRToolButtonSetHelpText (
    FR\_ToolButton btn,
    FS\_LPCWSTR helpText
);
```

**Description**

Sets the help text of the specified button.

**Parameter**

---

btn	[In] The button to which a tooltip is added.
-----	--

---

helpText	[In] The text to show.
----------	------------------------

---

**Return**

TRUE means successful, otherwise not.

**Head file reference**

fr\_barTempl.h: 82

**Related method**

[FRToolButtonSetLabelText](#)

**FRToolButtonSetIcon****Syntax**

```
FS_BOOL FRToolButtonSetIcon (
    FR\_ToolButton btn,
    FS\_DIBitmap bitmap,
    FRToolBar flyToolbar
);
```

**Description**

Sets a new bitmap for a toolbar button. The size of bitmap which to be set to a button is different between common toolbar mode and tabbed toolbar mode, with common mode, the size of bitmap is 24 \* 24, but 32 \* 32 with tabbed mode.

**Parameter**

---

btn	[In] The button whose icon is set.
-----	------------------------------------

---

bitmap	[In] The icon to set.
--------	-----------------------

---

flyToolbar	[In] If this button is used for fly toolbar, input the fly toolbar. Otherwise, set it to NULL.
------------	---

---

**Return**

TRUE means successful, otherwise not.

**Head file reference**

fr\_barTempl.h: 98

**Note:** The bitmap to set for a toolbar button will be maintained and released by the toolbar. So client can not release it.

**FRTadioButtonSetLabelText****Syntax**

```
FS_BOOL FRTadioButtonSetLabelText (
    FR_ToolButton btn,
    FS_LPCWSTR labelText
);
```

**Description**

Sets the label text of the specified button.

**Parameter**

btn	[In] The button whose label text is set.
labelText	[In] A pointer to a wide string buffer. The string buffer is read-only.

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_barTempl.h: 75

**Related method**[FRTadioButtonGetLabelText](#)[FRTadioButtonSetHelpText](#)**FRTadioButtonSetMarkedProc****Syntax**

```
FS_BOOL FRTadioButtonSetMarkedProc (
    FR_ToolButton btn,
    FRComputeMarkedProc proc
);
```

**Description**

Sets a [FRBnCheckProc](#) () associated with a button.

**Parameter**

---

btn	[In] The button whose <a href="#">FRBtnCheckProc()</a> is set.
proc	[In] A user-supplied procedure to call whenever Reader needs to know whether the specified button should be marked.

---

**Return**

[TRUE](#) means successful, otherwise not.

**Head file reference**

fr\_barTempl.h: 34

**Related method**

[FRToolButtonNew](#)

[FRToolButtonSetExecuteProc](#)

[FRToolButtonSetEnableProc](#)

[FRToolButtonSetClientData](#)

**Note:** The user-supplied data passed to FRBtnCheckProc() must be set if the callback FRBtnCheckProc() need. Using FRToolButtonSetClientDate() to set it.

## FR\_Touchup

### Description

### Definitions

#### Definitions summary

[FR\\_TEXT\\_OP\\_CREATE](#)

[FR\\_TEXT\\_OP\\_DELETE](#)

[FR\\_TEXT\\_OP\\_EDIT](#)

[FR\\_TEXT\\_OP\\_MERGE](#)

[FR\\_TEXT\\_OP\\_NONE](#)

[FR\\_TEXT\\_OP\\_POSTION\\_CHANGE](#)

[FR\\_TEXT\\_OP\\_RESIZE](#)

[FR\\_TEXT\\_OP\\_ROTATE](#)

[FR\\_TEXT\\_OP\\_SHEAR](#)

[FR\\_TEXT\\_OP\\_SPLIT](#)

[FR\\_TEXT\\_OP\\_ON\\_REDO](#)

[FR\\_TEXT\\_OP\\_ON\\_RELEASE](#)

[FR\\_TEXT\\_OP\\_ON\\_UNDO](#)

#### Definitions detail

##### [FR\\_TEXT\\_OP\\_CREATE](#)

#### Syntax

```
#define FR_TEXT_OP_CREATE 0x1
```

**Description**

**Group**

[TextOperations](#)

**Head file reference**

fr\_appExpT.h: 7179

## FR\_TEXT\_OP\_DELETE

**Syntax**

```
#define FR_TEXT_OP_DELETE 0x2
```

**Description**

**Group**

[TextOperations](#)

**Head file reference**

fr\_appExpT.h: 7182

## FR\_TEXT\_OP\_EDIT

**Syntax**

```
#define FR_TEXT_OP_EDIT 0x4
```

**Description**

**Group**

[TextOperations](#)

**Head file reference**

fr\_appExpT.h: 7185

## FR\_TEXT\_OP\_MERGE

**Syntax**

```
#define FR_TEXT_OP_MERGE 0x100
```

**Description**

**Group**

[TextOperations](#)

**Head file reference**

fr\_appExpT.h: 7203

## FR\_TEXT\_OP\_NONE

### Syntax

```
#define FR_TEXT_OP_NONE 0
```

### Description

### Group

[TextOperations](#)

### Head file reference

fr\_appExpT.h: 7176

## FR\_TEXT\_OP\_POSITION\_CHANGE

### Syntax

```
#define FR_TEXT_OP_POSITION_CHANGE 0x8
```

### Description

### Group

[TextOperations](#)

### Head file reference

fr\_appExpT.h: 7188

## FR\_TEXT\_OP\_RESIZE

### Syntax

```
#define FR_TEXT_OP_RESIZE 0x40
```

### Description

### Group

[TextOperations](#)

### Head file reference

fr\_appExpT.h: 7197

## FR\_TEXT\_OP\_ROTATE

### Syntax

```
#define FR_TEXT_OP_ROTATE 0x10
```

### Description

**Group**

[TextOperations](#)

**Head file reference**

fr\_appExpT.h: 7191

## FR\_TEXT\_OP\_SHEAR

**Syntax**

```
#define FR_TEXT_OP_SHEAR 0x20
```

**Description**

**Group**

[TextOperations](#)

**Head file reference**

fr\_appExpT.h: 7194

## FR\_TEXT\_OP\_SPLIT

**Syntax**

```
#define FR_TEXT_OP_SPLIT 0x80
```

**Description**

**Group**

[TextOperations](#)

**Head file reference**

fr\_appExpT.h: 7200

## FR\_TEXT\_OP\_ON\_REDO

**Syntax**

```
#define FR_TEXT_OP_ON_REDO 1
```

**Description**

**Group**

[TextOperationUndoRedoType](#)

**Head file reference**

fr\_appExpT.h: 7250

## FR\_TEXT\_OP\_ON\_RELEASE

**Syntax**

```
#define FR_TEXT_OP_ON_RELEASE 2
```

**Description****Group**

[TextOperationUndoRedoType](#)

**Head file reference**

fr\_appExpT.h: 7252

## FR\_TEXT\_OP\_ON\_UNDO

**Syntax**

```
#define FR_TEXT_OP_ON_UNDO 0
```

**Description****Group**

[TextOperationUndoRedoType](#)

**Head file reference**

fr\_appExpT.h: 7248

## Structures

### Structures summary

[FR\\_Text\\_Object\\_OP](#)

[FR\\_Text\\_Object\\_OP\\_Result](#)

### Structs detail

#### FR\_Text\_Object\_OP

**Syntax**

```
typedef struct __FR_Text_Object_OP__{
    FPD\_PageObject textObj,
    FS\_INT32 iOP,
    FS\_INT32 iIndexInPage,
    void* pVoid,
    FS\_FloatRect rcBBox,
    void* pPos,
    short iParentOp
}FR_Text_Object_OP;
```

**Description****Head file reference**

fr\_appExpT.h: 7214

**textObj**

**iOP**

**iIndexInPage**

**pVoid**

**rcBBox**

**pPos**

**iParentOp**

**FR\_Text\_Object\_OP\_Result**

**Syntax**

```
typedef struct __FR_Text_Object_OP_Result__{
    FS_INT32 iUuid,
    FS_INT32 iStatus,
    void* pUndoItem,
    void (*OnRedoUndo)(void* pData, FS_INT32 iType)
}FR_Text_Object_OP_Result;
```

**Description**

**Head file reference**

fr\_appExpT.h: 7262

**iUuid**

**iStatus**

**pUndoItem**

**(\*OnRedoUndo)(void\* pData, FS\_INT32 iType)**

## FR\_UIProgress

[\*\*Return from Used by\*\*](#)

**Description**

The [\*\*FR\\_UIProgress\*\*](#) object is referred to a progress bar. See [\*\*FRUIProgressCreate\*\*](#).

**Returned from**

[\*\*FRUIProgressCreate\*\*](#)

**FRUIProgressCreate2**

Used by

[\*\*FRUIProgressDestroy\*\*](#)  
[\*\*FRUIProgressDoCancel\*\*](#)  
[\*\*FRUIProgressGetCurrentValue\*\*](#)  
[\*\*FRUIProgressIsCancelled\*\*](#)  
[\*\*FRUIProgressPeekAndPump\*\*](#)  
[\*\*FRUIProgressSetCurrValue\*\*](#)  
[\*\*FRUIProgressSetRange\*\*](#)  
[\*\*FRUIProgressSetText\*\*](#)

## Functions

Functions summary

**FRUIProgressCreate**

Creates a new [FR\\_UIProgress](#) object.

**FRUIProgressCreate2**

Creates a new [FR\\_UIProgress](#) object.

**FRUIProgressDestroy**

Destroy the progress bar.

**FRUIProgressDoCancel**

To cancel the progress bar.

**FRUIProgressGetCurrentValue**

The current value of the progress bar.

**FRUIProgressIsCancelled**

Whether the progress bar is cancelled or not.

**FRUIProgressPeekAndPump**

Peek and pump.

**FRUIProgressSetCurrValue**

Set the current value of the progress bar.

**FRUIProgressSetRange**

Set the range of the progress bar.

**FRUIProgressSetText**

Set the text shown in the progress bar.

## Functions detail

**FRUIProgressCreate**

**Syntax**

```
FR_UIProgress FRUIProgressCreate (
    HWND hParent,
    FS\_BOOL bShowCancelButton
);
```

**Description**

Creates a new [FR\\_UIProgress](#) object.

**Parameter**


---

hParent	[In] The parent window handle of the progress bar.
bShowCancelButton	[In] Whether to show the cancel button on the progress bar.

---

**Return**

The [FR\\_UIProgress](#) object created.

**Head file reference**

fr\_appTempl.h: 2382

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRUIProgressCreate2****Syntax**

```
FR_UIProgress FRUIProgressCreate2 (
    HWND hParent,
    FS\_BOOL bShowCancelButton,
    FS\_BOOL bRevertFocus
);
```

**Description**

Creates a new [FR\\_UIProgress](#) object.

**Parameter**


---

hParent	[In] The parent window handle of the progress bar.
bShowCancelButton	[In] Whether to show the cancel button on the progress bar.
bRevertFocus	[In] Whether to revert the focus to the original focused window.

---

**Return**

The [FR\\_UIProgress](#) object created.

**Head file reference**

fr\_appTempl.h: 2471

**Since**

[SDK LATEEST VERSION > 7.3.1](#)

**FRUIProgressDestroy**

**Syntax**

```
void FRUIProgressDestroy (
    FR\_UIProgress UIProgress
);
```

**Description**

Destroy the progress bar.

**Parameter**

---

UIProgress	[In] The input progress bar.
------------	------------------------------

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 2451

**FRUIProgressDoCancel****Syntax**

```
FS_BOOL FRUIProgressDoCancel (
    FR\_UIProgress UIProgress,
    FS\_BOOL bKill
);
```

**Description**

To cancel the progress bar.

**Parameter**

---

UIProgress	[In] The input progress bar.
------------	------------------------------

---

---

bKill	[In] Whether to kill the progress bar.
-------	--

---

**Return**

[TRUE](#) means success, otherwise failure.

**Head file reference**

fr\_appTempl.h: 2460

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRUIProgressGetCurrentValue**

**Syntax**

```
FS_INT32 FRUIProgressGetCurrentValue (
    FR\_UIProgress UIProgress
);
```

**Description**

The current value of the progress bar.

**Parameter**

---

UIProgress	[In] The input progress bar.
------------	------------------------------

---

**Return**

Get the current value of the progress bar.

**Head file reference**

fr\_appTempl.h: 2442

**FRUIProgressIsCancelled****Syntax**

```
FS_BOOL FRUIProgressIsCancelled (
    FR\_UIProgress UIProgress
);
```

**Description**

Whether the progress bar is cancelled or not.

**Parameter**

---

UIProgress	[In] The input progress bar.
------------	------------------------------

---

**Return**

Whether the progress bar is cancelled or not.

**Head file reference**

fr\_appTempl.h: 2433

**FRUIProgressPeekAndPump****Syntax**

```
void FRUIProgressPeekAndPump (
    FR\_UIProgress UIProgress
);
```

**Description**

Peek and pump.

#### Parameter

---

UIProgress	[In] The input progress bar.
------------	------------------------------

---

#### Return

void.

#### Head file reference

fr\_appTempl.h: 2424

### FRUIProgressSetCurrValue

#### Syntax

```
void FRUIProgressSetCurrValue (
    FR\_UIProgress UIProgress,
    FS\_INT32 nPos
);
```

#### Description

Set the current value of the progress bar.

#### Parameter

---

UIProgress	[In] The input progress bar.
------------	------------------------------

---

---

#### nPos

[In] The current value of the progress bar.

---

#### Return

void.

#### Head file reference

fr\_appTempl.h: 2414

### FRUIProgressSetRange

#### Syntax

```
void FRUIProgressSetRange (
    FR\_UIProgress UIProgress,
    FS\_INT32 nLower,
    FS\_INT32 nUpper
);
```

#### Description

Set the range of the progress bar.

**Parameter**

---

UIProgress	[In] The input progress bar.
nLower	[In] The minimize value of the range.
nUpper	[In] The maximize value of the range.

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 2403

**FRUIProgressSetText****Syntax**

```
void FRUIProgressSetText (
    FR_UIProgress UIProgress,
    FS_LPCWSTR lpText
);
```

**Description**

Set the text shown in the progress bar.

**Parameter**

---

UIProgress	[In] The input progress bar.
lpText	[In] The text shown in the progress bar.

---

**Return**

void.

**Head file reference**

fr\_appTempl.h: 2393

**FR\_VariableText****Description****FR\_VariableText\_Iterator**

Description

## **FR\_VariableText\_Provider**

Description

## **FR\_VTLine**

Description

## **FR\_VTSecProps**

Description

## **FR\_VTSection**

Description

## **FR\_VTWord**

Description

## **FR\_VTWordPlace**

Description

## **FR\_VTWordProps**

Description

## **FR\_VTWordRange**

Description

## **FR\_WindowsDIB**

Description

Functions

Functions summary

[FRWindowsDIBNew](#)

Constructs a DIB that can be efficiently interact with Windows device. Currently the constructed DIB will always in [FS\\_DIB\\_Rgb](#) format.

#### [\*\*FRWindowsDIBDestroy\*\*](#)

Destroys the input DIB bitmap.

#### [\*\*FRWindowsDIBGetBitmapInfo\*\*](#)

Get Windows bitmap info structure. The result is a binary string that can be used at a *BITMAPINFO* structure.

#### [\*\*FRWindowsDIBLoadFromBuf\*\*](#)

Constructs a bitmap from existing data.

#### [\*\*FRWindowsDIBGetDDBBitmap\*\*](#)

Converts to device compatible bitmap.

#### [\*\*FRWindowsDIBLoadFromDDB\*\*](#)

Load DI bitmap from DDB. If *hDC* is NULL, system display DC will be used.

#### [\*\*FRWindowsDIBLoadFromFile\*\*](#)

Load DI bitmap from file. Unicode version. Without GDI+ support, we supports only BMP file. With GDI support, we support much more.

#### [\*\*FRWindowsDIBLoadFromFileII\*\*](#)

Load DI bitmap from file. Unicode version. Without GDI+ support, we supports only BMP file. With GDI support, we support much more.

#### [\*\*FRWindowsDIBLoadDIBitmap\*\*](#)

Load DI bitmap from file. Unicode version. Without GDI+ support, we supports only BMP file. With GDI support, we support much more.

#### [\*\*FRWindowsDIBGetDC\*\*](#)

Gets the DC.

#### [\*\*FRWindowsDIBGetWindowsBitmap\*\*](#)

Gets the bitmap.

#### [\*\*FRWindowsDIBLoadFromDevice\*\*](#)

Loads from a device.

#### [\*\*FRWindowsDIBSetToDevice\*\*](#)

Outputs to a device.

## Functions detail

### [\*\*FRWindowsDIBNew\*\*](#)

#### **Syntax**

```
FS_DIBitmap FRWindowsDIBNew (
    HDC hDC,
    FS\_INT32 width,
    FS\_INT32 height
);
```

#### **Description**

Constructs a DIB that can be efficiently interact with Windows device. Currently the constructed DIB will always in [FS\\_DIB\\_Rgb](#) format.

#### **Parameter**

---

hDC	[In] The input windows device context.
-----	--

---

---

width	[In] The bimtap width.
-------	------------------------

---

height	[In] The bitmap height.
--------	-------------------------

---

**Return**

A DIB.

**Head file reference**

fr\_sysTempl.h: 144

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRWindowsDIBDestroy

**Syntax**

```
void FRWindowsDIBDestroy (
    FS\_DIBitmap pBitmap
);
```

**Description**

Destroys the input DIB bitmap.

**Parameter**

---

pBitmap	[In] The input DIB bitmap.
---------	----------------------------

---

**Return**

void.

**Head file reference**

fr\_sysTempl.h: 157

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRWindowsDIBGetBitmapInfo

**Syntax**

```
void FRWindowsDIBGetBitmapInfo (
    FS\_DIBitmap pBitmap,
    FS\_BytString* outInfo
);
```

**Description**

Get Windows bitmap info structure. The result is a binary string that can be used at a *BITMAPINFO* structure.

**Parameter**

---

pBitmap	[In] The input DIB bitmap.
outInfo	[Out] It receives the windows bitmap info structure for the DIB.

---

**Return**

void.

**Head file reference**

fr\_sysTempl.h: 167

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRWindowsDIBLoadFromBuf****Syntax**

```
FS_DIBitmap FRWindowsDIBLoadFromBuf (
    BITMAPINFO* pbmi,
    void* pData
);
```

**Description**

Constructs a bitmap from existing data.

**Parameter**

---

pbmi	[In] The windows bitmap info structure.
pData	[In] The bitmap data.

---

**Return**

A bitmap from existing data.

**Head file reference**

fr\_sysTempl.h: 179

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRWindowsDIBGetDDBitmap****Syntax**

```
HBITMAP FRWindowsDIBGetDDBitmap (
```

---

```
FS_DIBitmap pBitmap,
HDC hDC
);
```

**Description**

Converts to device compatible bitmap.

**Parameter**


---

pBitmap	[In] The input DIB.
---------	---------------------

---

hDC	[In] The input DC.
-----	--------------------

---

**Return**

A device dependent bitmap compatible with the input DC.

**Head file reference**

fr\_sysTempl.h: 190

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRWindowsDIBLoadFromDDB****Syntax**

```
FS_DIBitmap FRWindowsDIBLoadFromDDB (
    HDC hDC,
    HBITMAP hBitmap,
    FS_DWORD* pPalette,
    FS_DWORD size
);
```

**Description**

Load DI bitmap from DDB. If *hDC* is NULL, system display DC will be used.

**Parameter**


---

hDC	[In] The input DC.
-----	--------------------

---

hBitmap	[In] The input device dependent bitmap.
---------	---

---

pPalette	[In] The bmp's palette, applicable to 1bppRgb and 8bppRgb formats.
----------	--

---

size	[In] The palette's size.
------	--------------------------

---

**Return**

A DIB.

**Head file reference**

fr\_sysTempl.h: 201

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRWindowsDIBLoadFromFile****Syntax**

```
FS_DIBitmap FRWindowsDIBLoadFromFile (
    FS\_LPCWSTR filename
);
```

**Description**

Load DI bitmap from file. Unicode version. Without GDI+ support, we supports only BMP file. With GDI support, we support much more.

**Parameter**

---

filename	[In] The input full file path.
----------	--------------------------------

---

**Return**

A DIB.

**Head file reference**

fr\_sysTempl.h: 214

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FRWindowsDIBLoadFromFileII****Syntax**

```
FS_DIBitmap FRWindowsDIBLoadFromFileII (
    FS\_LPCSTR filename
);
```

**Description**

Load DI bitmap from file. Unicode version. Without GDI+ support, we supports only BMP file. With GDI support, we support much more.

**Parameter**

---

filename	[In] The input full file path.
----------	--------------------------------

---

**Return**

A DIB.

**Head file reference**

fr\_sysTempl.h: 225

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

FRWindowsDIBLoadDIBitmap

**Syntax**

```
FS_DIBitmap FRWindowsDIBLoadDIBitmap (
    FR\_WINDIB\_Open\_Args args
);
```

**Description**

Load DI bitmap from file. Unicode version. Without GDI+ support, we supports only BMP file. With GDI support, we support much more.

**Parameter**

---

args	[In] The input full file path or memory.
------	--

---

**Return**

A DIB.

**Head file reference**

fr\_sysTempl.h: 236

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

FRWindowsDIBGetDC

**Syntax**

```
HDC FRWindowsDIBGetDC (
    FS\_DIBitmap pBitmap
);
```

**Description**

Gets the DC.

**Parameter**

---

pBitmap	[In] The input DIB bitmap.
---------	----------------------------

---

**Return**

The DC.

**Head file reference**

fr\_sysTempl.h: 247

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRWindowsDIBGetWindowsBitmap

**Syntax**

```
HBITMAP FRWindowsDIBGetWindowsBitmap (
    FS\_DIBitmap pBitmap
);
```

**Description**

Gets the bitmap.

**Parameter**

---

pBitmap	[In] The input DIB bitmap.
---------	----------------------------

---

**Return**

The bitmap handle.

**Head file reference**

fr\_sysTempl.h: 257

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FRWindowsDIBLoadFromDevice

**Syntax**

```
void FRWindowsDIBLoadFromDevice (
    FS\_DIBitmap pBitmap,
    HDC hDC,
    FS\_INT32 left,
    FS\_INT32 top
);
```

**Description**

Loads from a device.

**Parameter**

pBitmap	[In] The input DIB bitmap.
hDC	[In] The input windows device context.
left	[In] The x-coordinate in the windows DC.
top	[In] The y-coordinate in the windows DC.

**Return**

The bitmap.

**Head file reference**

fr\_sysTempl.h: 267

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FRWindowsDIBSetToDevice****Syntax**

```
void FRWindowsDIBSetToDevice (
    FS\_DIBitmap pBitmap,
    HDC hDC,
    FS\_INT32 left,
    FS\_INT32 top
);
```

**Description**

Outputs to a device.

**Parameter**

pBitmap	[In] The input DIB bitmap.
hDC	[In] The input windows device context.
left	[In] The x-coordinate in the windows DC.
top	[In] The y-coordinate in the windows DC.

**Return**

void.

**Head file reference**

fr\_sysTempl.h: 280

Since

[SDK LATEST VERSION > 1.0](#)

## FRMS\_Common

Description

## FRMS\_Decryption

Description

## FRMS\_Encryption

Description

# PD Layer

---

### Object list

The Acrobat Support (AS) layer of the core API provides a variety of utility methods, including platform-independent memory allocation and fixed-point math utilities. In addition, it allows plug-ins to replace low-level file system routines used by Acrobat (including read, write, reopen, remove file, rename file, and other directory operations). This enables Acrobat to be used with other file systems, such as on-line systems. Several AS methods return error codes rather than raising exceptions on errors. This is because these methods are called at a low level where exception handling would be inconvenient and expensive.

## Object List

### General

### FDF Document

The underlying FDF (Form Data Format) representation of a document. The file format used for interactive form data.

### FDRM\_PDFSecurityHandler

### FOFD\_ACTION

### FOFD\_ActionArea

[FOFD\\_ActionGoto](#)

[FOFD\\_ActionGotoA](#)

[FOFD\\_ActionRegion](#)

[FOFD\\_ACTIONS](#)

[FOFD\\_ANNOT](#)

[FOFD\\_Bookmark](#)

[FOFD\\_Bookmarks](#)

[FOFD\\_CREATOR](#)

[FOFD\\_CRYPTODICT](#)

[FOFD\\_CryptoHandler](#)

[FOFD\\_DEST](#)

[FOFD\\_DIBAttribute](#)

[FOFD\\_Document](#)

The underlying OFD representation of a document.

[FOFD\\_DriverDevice](#)

[FOFD\\_FILEPACKAGE](#)

[FOFD\\_FileStream](#)

[FOFD\\_OBJECT](#)

The underlying OFD representation of a object.

[FOFD\\_OUTLINE](#)

[FOFD\\_PAGE](#)

The underlying OFD representation of a page.

## **FOFD\_PARSER**

## **FOFD\_Path**

## **FOFD\_PauseHandler**

An object preparing a simple pause instance.

## **FOFD\_PERMISSIONS**

## **FOFD\_ProgressiveRenderer**

## **FOFD\_RenderContext**

## **FOFD\_RenderDevice**

## **FOFD\_RenderOptions**

## **FOFD\_SecurityHandler**

## **FOFD\_SIGNATURE**

The underlying OFD representation of a Signature.

## **FOFD\_VPREFERENCES**

## **FOFD\_WRITEPAGE**

The underlying OFD representation of a page.

## **FOFD\_WRITESIGNATURE**

The underlying OFD representation of a Signature.

## **FPD\_AAction**

Additional-action extends a set of events that can trigger the execution of actions. It is specified in /AA entry in a annotation, page object, interactive form or document catalog. See [FPDAActionNew](#) , [FPDAActionDestroy](#) , [FPDFFormFieldGetAdditionalAction](#) , [FPDFFormControlGetAdditionalAction](#) .

## **FPD\_Action**

Actions are what happens when a user clicks on a link or bookmark.

## **FPD\_ActionFields**

FPD\_ActionFields represents the /Fields entry in submit-form actions dictionary. We can use a FPD\_ActionFields object to get field's information easily. See [FPDACTIONFIELDSNEW](#) , [FPDACTIONFIELDSDESTROY](#) , [FPDACTIONGETWIDGETS](#) .

## **FPD\_AllStates**

No document exists.

## **FPD\_Annot**

An annotation on a page in a PDF file. See [FPDANNOTNEW](#) , [FPDANNOTDESTROY](#) , [FRANNOTGETPDAANNOT](#) , [FPDANNOTLISTGETAT](#) .

## **FPD\_AnnotList**

An annotation accessor. See [FPDANNOTLISTNEW](#) , [FPDANNOTLISTDESTROY](#) .

## **FPD\_Array**

A FPD\_Array is one-dimensional collections of objects accessed by a numeric index. Array indexes are zero-based. An array's elements may be any combination of the FPD\_Object types.

## **FPD\_BackgroundDrawHandler**

PDF background drawing interface. Use for holding a [FPD\\_BackgroundDraw](#) .

## **FPD\_Bookmark**

A bookmark corresponds to an outline object in a PDF document ( *see Section 8.2.2, Document Outline, in the PDF Reference* ).

## **FPD\_Boolean**

FPD\_Boolean objects can have a value of [TRUE](#) or [FALSE](#) .

## **FPD\_CIDFont**

The CIDFont. A CIDFont is designed to contain a large number of glyph procedures and is used for languages such as Chinese, Japanese, or Korean. Instead of being accessed by a name, each glyph procedure is accessed by an integer known as a character identifier or CID. Instead of a font encoding, CIDFonts use a CMap to define the mapping from character codes to a font number and a character selector.

## **FPD\_CIDUtil**

An utility class for CID processing. See [FPDCIDUtilIsVerticalJapanCID](#) .

## **FPD\_ClipPath**

A clip path. See [FPDClipPathNew](#) , [FPDClipPathDestroy](#) .

## **FPD\_Color**

PDF color. see *CHAPTER 4.5 in PDF reference* . See [FPDColorNew](#) , [FPDColorDestroy](#) .

## **FPD\_ColorSpace**

The color space used in pdf color system. see *CHAPTER 4.5 in PDF reference* . See [FPDColorSpaceLoad](#) , [FPDColorSpaceReleaseCS](#) .

## **FPD\_ColorState**

The color state for rendering. See [FPDColorStateNew](#) , [FPDColorStateDestroy](#) .

## **FPD\_ConnectedInfo**

The connected PDF info object can get connected PDF document ID and version ID etc.  
Typical usage of this object is to get and set connected PDF info.

## **FPD\_ContentGenerator**

PDF Page or Form content generator.

## **FPD\_ContentMark**

The content marks for tagged pdf. See [FPDContentMarkNew](#) , [FPDContentMarkDestroy](#) .

## **FPD\_ContentMarkItem**

The content mark item for tagged pdf. See [FPDContentMarkItemNew](#) , [FPDContentMarkItemDestroy](#) .

## **FPD\_Creator**

PDF Creator: creating PDF file from a document object. See [FPDCreatorNew](#) , [FPDCreatorDestroy](#) .

## **FPD\_CreatorOption**

The [FPD\\_CREATOROPTION](#) object is used to decode or encode the stream progressively. It is returned by [FPDCreatorSetOption](#) and can be released by [FPDCreatorReleaseOption](#).

## **FPD\_CryptoHandler**

Abstract class for PDF cryptographic operations (encryption and decryption). This class works with security handler which provides algorithm and key info.

## **FPD\_DefaultAppearance**

Default appearance interpreter, serves for DA entry in form field dictionary.

## **FPD\_Dest**

Corresponding to a PDF Dest array.

## **FPD\_Dictionary**

A FPD\_Dictionary is an associative table whose elements are pairs of objects:

## **FPD\_DocJSActions**

A JavaScript action causes a script to be complied and executed by the JavaScript terpreter. See [FPDDocJSActionsNew](#) , [FPDDocJSActionsDestroy](#) .

## **FPD\_Document**

The underlying PDF representation of a document.

## **FPD\_EnumPageHandler**

A handler to enumerate the count of pages for a document. See [FPDDocNewEnumPageHandler](#) , [FPDDocDeleteEnumPageHandler](#) .

## **FPD\_EPUB**

## **FPD\_FileSpec**

PDF file specification object. A FPD\_FileSpec corresponds to the PDF file specification object (see Section 3.10, File Specifications, in the PDF Reference). It is used to specify a file in an action.

## **FPD\_Font**

The [FPD\\_Font](#) object is used to manipulate the PDF font. See [FPDFFontNew](#) , [FPDFFontDestroy](#) .

## **FPD\_FontEncoding**

Single byte font encoding used by T1 or TT fonts. See [FPDFFontEncodingNew](#) , [FPDFFontEncodingDestroy](#) .

## **FPD\_Form**

A [FPD\\_Form](#) is a self-contained set of graphics operators that is used when a particular graphic is drawn more than once in a document. It corresponds to a PDF XObject (*see Section 4.9, Form XObjects, the PDF Reference*) . See [FPDFFormNew](#) , [FPDFFormDestroy](#) .

## **FPD\_FormControl**

A FPD\_FormControl is an appearance controller, it used to draw an annotation's presentation on the page or set the appearance data to change an annotation's appearance. See [FPDInterFormNewControl](#) , [FPDInterFormGetControl](#) , [FPDInterFormGetPageControl](#) , [FPDInterFormGetControlAtPoint](#) , [FPDInterFormGetControlByDict](#) , [FPDInterFormDeleteControl](#) .

## **FPDFormField**

A field in a interactive form. See [FPDInterFormNewField](#) , [FPDInterFormGetField](#) , [FPDInterFormGetFieldByDict](#) , [FPDInterFormDeleteField](#) , [FPDInterFormDeleteField2](#) .

## **FPD\_FormNotify**

A object representing a form notify handler.

## **FPD\_FormObject**

This object corresponds to a PDF form XObject ( *see Section 4.9, Form XObjects, in the PDF Reference* ).

## **FPD\_FT\_Face**

FreeType root face class structure. A face object models a typeface in a font file.

## **FPD\_Function**

No document exists.

## **FPD\_FXFont**

The [FPD\\_FXFont](#) object is used to manipulate the Foxit GE font. See [FPDFFontGetFXFont](#) .

## **FPD\_FXFontEncoding**

It works with a font to translate character codes into glyph indices in that font. It also deals with Unicode mapping (if supported). See [FPDFXFontEncodingNew](#) , [FPDFXFontEncodingDestroy](#) .

## **FPD\_FxgeDevice**

Foxit own FXGE rendering device, using the Foxit FXGE device driver. Supporting output to following formats:

- 8bppMask: the output will be the alpha channel.
- 8bppRgb without palette: the output will be grayscale.
- Rgb, Rgb32: normal RGB output.
- Argb: RGB with alpha channel output.

## **FPD\_GeneralState**

The general state for page rendering. See [FPDGeneralStateNew](#) , [FPDGeneralStateDestroy](#) .

## **FPD\_GraphState**

The graph state for graphic rendering.

## **FPD\_IconFit**

A FPD\_IconFit represents a set of icon that specifying how to display the widget otation's icon within its annotation rectangle. See [FPDIconFitNew](#) , [FPDIconFitDestroy](#) .

## **FPD\_Image**

Holding image(sampled image) data . see *CHAPTER 4.8 in PDF reference* .

## **FPD\_ImageObject**

A FPD\_ImageObject is a sampled image or image mask, and corresponds to a PDF Image resource (see *Stencil Masking in Section 4.8, Images, in the PDF Reference* ).

## **FPD\_InlineImages**

A FPD\_InlineImages is an image whose data is stored in the page description's contents stream instead of being stored as an image resource (see FPD\_ImageObject).

FPD\_InlineImages is a subclass of FPD\_PageObject and corresponds to the PDF inline image operator ( see *Section 4.8.6, In-Line Images, in the PDF Reference* ).

## **FPD\_InterForm**

A PDF interactive form (AcroForm) object.

## **FPD\_Link**

A FPD\_Link corresponds to a link annotation(see Sections 8.4.5, Annotation Types, the PDF Reference). See [FPDLinkNew](#) , [FPDLinkDestroy](#) [FPDLinkGetLinkAtPoint](#) , [FPDLinkGetLink](#) .

## **FPD\_LinkExtract**

A link extractor for accessing links. See [FPDLinkExtractNew](#) , [FPDLinkExtractDestroy](#) .

## **FPD\_LWinParam**

A FPD\_LWinParam represents a set of parameter on Windows desktop to launch an application or open or print a document. It is used by launch actions. See [FPDLWinParamNew](#) , [FPDLWinParamDestroy](#) , [FPDACTIONGetWinParam](#) .

## **FPD\_MediaPlayer**

A media player information object. See [FPDMediaNew](#) , [FPDMediaNewFromDict](#) , [FPDMediaDestroy](#) .

## **FPD\_MeshStream**

No document exists. See [FPDMeshStreamNew](#) , [FPDMeshStreamDestroy](#) .

## **FPD\_Name**

A FPD\_Name is a sequence of non-white space characters. In code, a name is preceded by the forward slash (/) character indicating that it is a string literal, for example: /AName. See Section 3.2.4 in the PDF Reference for details.

## **FPD\_NameTree**

The dictionary used to store all of the Named Destinations in a PDF file.

## **FPD\_Null**

There is only one **NULL** object, which is used to fill empty or uninitialized positions in arrays or dictionaries. See Section 3.2.8 in the PDF Reference for details.

## **FPD\_Number**

FPD\_Number may be specified by signed or unsigned constants. See Section 3.2.2 in the PDF Reference for details.

## **FPD\_ObjArchiveLoader**

PDF object archive loader class. See [CFPDObjArchiveLoaderNew](#) , [CFPDObjArchiveLoaderDestroy](#) .

## **FPD\_ObjArchiveSaver**

PDF object archive saver class. See [CFPDObjArchiveSaverNew](#) , [CFPDObjArchiveDestroy](#) .

## **FPD\_Object**

A [FPD\\_Object](#) is a general object in a PDF file, which may be of any object type. This is the abstract class for all PDF syntax objects.

## **FPD\_OCContext**

A FPD\_OCContext is an object that keeps track the on/off states of all of the OCGs a document.

## **FPD\_OCContextHandler**

PDF optional content context interface. Used for holding a [FPD\\_OCContextCallBack](#) .

## **FPD\_OCGroup**

A FPD\_OCGroup represents a named object whose state can be toggled in a user terface to affect changes in visibility of content.

## **FPD\_OCGroupSet**

A set of mutually exclusive OCGs. See [FPDOCGroupSetNew](#) , [FPDOCGroupSetDestroy](#) , [FPDOCGroupSetIsSubGroupSet](#) , [FPDOCGroupSetGetGroup](#) , [FPDOCGroupSetFindGroup](#) .

## **FPD\_OCNotify**

Optional Content Notification Interface. See [FPDOCNotifyFPD\\_OCNotifyNew](#) , [FPDOCNotifyFPD\\_OCNotifyDestroy](#) , [FPDOCPropertiesAddOCNotify](#) , [FPDOCPropertiesRemoveOCNotify](#) .

## **FPD\_OCProperties**

The optional content properties dictionary for a document. See [FPDOCPropertiesNew](#) , [FPDOCPropertiesDestroy](#) , [FPDOCPropertiesCountConfigs](#) , [FPDOCPropertiesGetConfig](#) , [FPDOCPropertiesAddOCNotify](#) , [FPDOCPropertiesRemoveOCNotify](#) .

## **FPD\_Page**

A [FPD\\_Page](#) is a page in a document, corresponding to the PDF Page object (see Page Objects in Section 3.6.2, Page Tree, in the PDF Reference).

## **FPD\_PageArchiveLoader**

PDF page archive loader class. See [CFPDPageArchiveLoaderNew](#) , [CFPDPageArchiveLoaderDestroy](#) .

## **FPD\_PageArchiveSaver**

PDF page archive saver class. See [CFPDPageArchiveSaverNew](#) , [CFPDPageArchiveSaverDestroy](#) .

## **FPD\_PageObject**

A [FPD\\_PageObject](#) is the abstract superclass of page objects.

## **FPD\_PageRenderCache**

Store all dispensable items used when rendering a page. See [FPD\\_BackgroundDraw](#) Note:  
Currently we have image cache only.

## **FPD\_ParseOptions**

The page parsing options.

## **FPD\_Parser**

## **FPD\_Path**

FPD path. A [FPD\\_Path](#) is a data container for a [FPD\\_PathObject](#) which is a graphic object representing a path in a page description. See [FPDPathNew](#) , [FPDPathDestroy](#) .

## **FPD\_PathObject**

A FPD\_PathObject is a graphic object (a subclass of FPD\_PageObject) representing a path in a page description.

## **FPD\_Pattern**

The abstract class for tiling pattern and shading pattern. Has no "new" functions. see  
*CHAPTER 4.6 in PDF reference* . See [FPDPatternDestroy](#) .

## **FPD\_ProgressiveEncryptHandler**

The [FPD\\_ProgressiveEncryptHandler](#) object is used to encrypt the file progressively. It is returned by [FPDCreatorSetProgressiveEncryptHandler](#) and can be released by [FPDCreatorReleaseProgressiveEncryptHandler](#) .

## **FPD\_ProgressiveRender**

The PDF progressive renderer.

## **FPD\_ProgressiveSearch**

A progressive search facility for stream-based text searching inside a single page. See [FPDProgressiveSearchNew](#) , [FPDProgressiveSearchDestroy](#) .

## **FPD\_Reference**

A FPD\_Reference is a additional type of object:  
Objects may be labeled so that they can be referred to by other objects. A labeled object is called an indirect object.

## **FPD\_RenderContext**

Context for rendering a PDF page or a list of page objects.

## **FPD\_RenderDevice**

The base function set for render devices.

## **FPD\_RenderOptions**

Page rendering options. see CHAPTER 6 in PDF reference. See [FPDRenderOptionsNew](#) , [FPDRenderOptionsDestroy](#) .

## **FPD\_Rendition**

A Rendition object. See [FPDRenditionNew](#) , [FPDRenditionNewFromDict](#) , [FPDRenditionDestroy](#) .

## **FPD\_ShadingObject**

A FPD\_PathObject is a graphic object (a subclass of FPD\_PageObject) representing a smooth shading in a page description.

## **FPD\_ShadingPattern**

PDF shading pattern. see *CHAPTER 4.6.3 in PDF reference* .

## **FPD\_Stream**

A FPD\_Stream is a sequence of characters that can be read a portion at a time. Streams are used for objects with large amounts of data, such as images, page content, or private data a plug-in creates. A stream consists of these elements, which are listed in their relative order in the stream object, starting at the beginning. See Section 3.2.7 in the PDF Reference for a description of the stream object.

## **FPD StreamAcc**

Accessor of stream object: [FPD StreamAcc](#) depends on a stream, it maintains a buffer for the data of a stream. This buffer may be temporary, if the stream is encoded by one or more data encoders. [FPD StreamAcc](#) doesn't decode any image encoders. See [FPDStreamAccNew](#) , [FPDStreamAccDestroy](#) .

## **FPD StreamFilter**

Data filter created for accessing PDF stream data. See [FPDStreamFilterDestroy](#) .

## **FPD StreamWrite**

stream writing interface.

## **FPD String**

A FPD\_String is a sequences of characters, enclosed in parentheses. *See Section 3.2.3 in the PDF Reference* for details.

## **FPD SubstFont**

Substitution font. See [FPDSubstFontNew](#) , [FPDSubstFontDestroy](#) .

## **FPD TextObject**

A FPD\_TextObject is a graphic object (a subclass of FPD\_PageObject) representing one or more character strings on a page.

## **FPD TextPage**

Text page for PDF text processing. See [FPDTextNew](#) , [FPDTextDestroy](#) .

## **FPD TextPageFind**

To find a text in a page. See [FPDTextPageFindNew](#) , [FPDTextPageFindDestroy](#) .

## **FPD TextState**

The text state for page rendering. See [FPDTextStateNew](#) , [FPDTextStateDestroy](#) .

## **FPD TilingPattern**

The tiling pattern. see *CHAPTER 4.6.2 in PDF reference* .

## **FPD TrueTypeFont**

The True-Type font.

## **FPD\_Type1Font**

The Type1 font.

## **FPD\_Type3Char**

Type3 character information. See [FPDType3CharNew](#) , [FPDType3CharDestroy](#) .

## **FPD\_Type3Font**

The Type3 font. Type 3 fonts do not have the ability to provide a base font with more than one encoding. For each Type 3 font, there is only one encoding. This encoding is completely specified in the PDF file; there are no shortcuts as there are for other fonts. *See Section 5.7, Font Descriptors, in the PDF Reference* for a discussion of font descriptors.

## **FPD\_UnencryptedWrapperCreator**

## **FPD\_WindowsDevice**

The rendering device based on Windows device driver.

## **FPD\_Wrapper20Creator**

PDF Wrapper Creator: creating a wrapper 2.0 PDF file to an existing PDF file. See [FPDWrapper20CreatorNew](#) , [FPDWrapper20CreatorDestroy](#) .

## **FPD\_WrapperCreator**

PDF Wrapper Creator: creating a wrapper PDF file to an existing PDF file. See [FPDWrapperCreatorNew](#) , [FPDWrapperCreatorDestroy](#) .

## **FPD\_WrapperDoc**

## **FRMS\_Template**

A [FRMS\\_Template](#) is an object that represents the rights policy template of RMS.

# **General**

## **Description**

## **Definitions**

### **Definitions summary**

**FPD\_CONTENT\_NOT\_PARSED**

The page is not parsed.

**FPD\_CONTENT\_PARSED**

The page is parsed.

**FPD\_CONTENT\_PARSING**

The page is parsing.

**FPD\_2TXT\_AUTO\_ROTATE**

Auto rotating.

**FPD\_2TXT\_AUTO\_WIDTH**

Auto width.

**FPD\_2TXT\_INCLUDE\_INVISIBLE**

Whether to include invisible texts.

**FPD\_2TXT\_KEEP\_COLUMN**

Keep column.

**FPD\_2TXT\_USE\_OCR**

Whether to use OCR.

**FPD\_TRANS\_GROUP**

Whether a transparency group is present.

**FPD\_TRANS\_ISOLATED**

Whether a transparency group is isolated. For isolated groups, a separate level device is required.

**FPD\_TRANS\_KNOCKOUT**

Whether a transparency group is knockout. For non-knockout groups, composition is required for all objects inside.

**FRMS\_ACTIVATE\_CREDENTIAL****FRMS\_ACTIVATE\_MACHINE****FRMS\_NEEDS\_CREDENTIAL\_ACTIVATION****FRMS\_NEEDS\_MACHINE\_ACTIVATION**

## Definitions detail

### FPD\_CONTENT\_NOT\_PARSED

**Syntax**

```
#define FPD_CONTENT_NOT_PARSED 0
```

**Description**

The page is not parsed.

**Group**

[FPDContentParsingState](#)

**Head file reference**

fpd\_pageExpT.h: 166

### FPD\_CONTENT\_PARSED

**Syntax**

```
#define FPD_CONTENT_PARSED 2
```

**Description**

The page is parsed.

**Group**

[FPDContentParsingState](#)

**Head file reference**

fpd\_pageExpT.h: 170

## FPD\_CONTENT\_PARSING

**Syntax**

```
#define FPD_CONTENT_PARSING 1
```

**Description**

The page is parsing.

**Group**

[FPDContentParsingState](#)

**Head file reference**

fpd\_pageExpT.h: 168

## FPD\_2TXT\_AUTO\_ROTATE

**Syntax**

```
#define FPD_2TXT_AUTO_ROTATE 1
```

**Description**

Auto rotating.

**Group**

[FPDOSTextExtractingFlags](#)

**Head file reference**

fpd\_pageExpT.h: 184

## FPD\_2TXT\_AUTO\_WIDTH

**Syntax**

```
#define FPD_2TXT_AUTO_WIDTH 2
```

**Description**

Auto width.

**Group**

[FPDOSTextExtractingFlags](#)**Head file reference**

fpd\_pageExpT.h: 186

**FPD\_2TXT\_INCLUDE\_INVISIBLE****Syntax**

#define FPD\_2TXT\_INCLUDE\_INVISIBLE 16

**Description**

Whether to include invisible texts.

**Group**[FPDOSTextExtractingFlags](#)**Head file reference**

fpd\_pageExpT.h: 192

**FPD\_2TXT\_KEEP\_COLUMN****Syntax**

#define FPD\_2TXT\_KEEP\_COLUMN 4

**Description**

Keep column.

**Group**[FPDOSTextExtractingFlags](#)**Head file reference**

fpd\_pageExpT.h: 188

**FPD\_2TXT\_USE\_OCR****Syntax**

#define FPD\_2TXT\_USE\_OCR 8

**Description**

Whether to use OCR.

**Group**[FPDOSTextExtractingFlags](#)**Head file reference**

fpd\_pageExpT.h: 190

## FPD\_TRANS\_GROUP

### Syntax

```
#define FPD_TRANS_GROUP 0x0100
```

### Description

Whether a transparency group is present.

### Group

[FPDTransparencyAttrFlags](#)

### Head file reference

fpd\_pageExpT.h: 148

## FPD\_TRANS\_ISOLATED

### Syntax

```
#define FPD_TRANS_ISOLATED 0x0200
```

### Description

Whether a transparency group is isolated. For isolated groups, a separate level device is required.

### Group

[FPDTransparencyAttrFlags](#)

### Head file reference

fpd\_pageExpT.h: 150

## FPD\_TRANS\_KNOCKOUT

### Syntax

```
#define FPD_TRANS_KNOCKOUT 0x0400
```

### Description

Whether a transparency group is knockout. For non-knockout groups, composition is required for all objects inside.

### Group

[FPDTransparencyAttrFlags](#)

### Head file reference

fpd\_pageExpT.h: 152

## FRMS\_ACTIVATE\_CREDENTIAL

### Syntax

```
#define FRMS_ACTIVATE_CREDENTIAL 0x02
```

**Description**

**Group**

[FRMS\\_ACTIVATION\\_FLAGS](#)

**Head file reference**

frmsExpT.h: 44

## FRMS\_ACTIVATE\_MACHINE

**Syntax**

```
#define FRMS_ACTIVATE_MACHINE 0x01
```

**Description**

**Group**

[FRMS\\_ACTIVATION\\_FLAGS](#)

**Head file reference**

frmsExpT.h: 41

## FRMS\_NEEDS\_CREDENTIAL\_ACTIVATION

**Syntax**

```
#define FRMS_NEEDS_CREDENTIAL_ACTIVATION 0x8004CF3E
```

**Description**

**Group**

[FRMS\\_ERR\\_CODES](#)

**Head file reference**

frmsExpT.h: 54

## FRMS\_NEEDS\_MACHINE\_ACTIVATION

**Syntax**

```
#define FRMS_NEEDS_MACHINE_ACTIVATION 0x8004CF3D
```

**Description**

**Group**

[FRMS\\_ERR\\_CODES](#)

**Head file reference**

frmsExpT.h: 51

## TypeDefs

### TypeDefs summary

#### [FRMSHRESULT](#)

### TypeDefs detail

#### FRMSHRESULT

##### Syntax

```
typedef long FRMSHRESULT;
```

##### Description

##### Head file reference

frmsExpT.h: 61

## Callbacks

### Callbacks summary

#### [FRMSCommonOnWillInitSecureEnvironment](#)

It is called when application will init the RMS secure environment. One application should init the RMS secure environment only once.

#### [FRMSDecryptionCusAuth](#)

It is called when application is authorizing and user wants to do additional authorization.

### Callbacks detail

#### FRMSCommonOnWillInitSecureEnvironment

##### Syntax

```
typedef void (*FRMSCommonOnWillInitSecureEnvironment)(  
    FS_LPVVOID clientData,  
    FS_LPWSTR pwszMachineCert,  
    FS_LPWSTR pwszManifest  
>);
```

##### Description

It is called when application will init the RMS secure environment. One application should init the RMS secure environment only once.

##### Parameter

---

clientData	[In] The user-supplied data.
pwszMachineCert	[In] The machine certificate used to init the RMS secure environment.

---

---

pwszManifest	[In] The manifest certificate used to init the RMS secure environment.
--------------	--

---

**Return**

void

**Head file reference**

frmsExpT.h: 89

**Group**[FRMS\\_CommonEventCallbacksRec](#)**FRMSDecryptionCusAuth****Syntax**

```
typedef FS_BOOL (*FRMSDecryptionCusAuth)(
    FS_LPVVOID clientData,
    FPD_Document doc,
    FS_LPWSTR pwszSIL,
    FS_LPWSTR pwszEULChain
);
```

**Description**

It is called when application is authorizing and user wants to do additional authorization.

**Parameter**


---

clientData	[In] The user-supplied data.
------------	------------------------------

---

doc	[In] The document app is authorizing.
-----	---------------------------------------

---

pwszSIL	[In] The SIL associated with the document.
---------	--

---

pwszEULChain	[In] The EUL app uses to do the authorization.
--------------	--

---

**Return**

TRUE means the additional authorization is done successfully. Once a additional authorizationis done unsuccessfully, the internal authorization will be terminated and the application will fail to open the document.

**Head file reference**

frmsExpT.h: 216

**Group**[FRMS\\_DecryptionCusAuthCallbacksRec](#)

## FDF\_Document

### [Return from Used by](#)

#### Description

The underlying FDF (Form Data Format) representation of a document. The file format used for interactive form data. An FDF file is structured in essentially the same way as a PDF file but contains only those elements required for the export and import of interactive form and annotation data. It consists of three required elements and one optional element: For more information, see Section 8.6 in the PDF Reference". The FDF document can be created from scratch, or parsed from a file or memory block. See [FPDFDFDocNew](#) , [FPDFDFDocOpenFromFile](#) , [FPDFDFDocParseMemory](#) .

#### Returned from

[FPDInterFormExportToFDF](#)  
[FPDInterFormExportToFDF2](#)  
[FPDFDFDocNew](#)  
[FPDFDFDocOpenFromFile](#)  
[FPDFDFDocParseMemory](#)

#### Used by

[FPDArrayAddReference2ToFDFDoc](#)  
[FPDArrayAddReferenceToFDFDoc](#)  
[FPDDictionarySetAtReference2ToFDFDoc](#)  
[FPDDictionarySetAtReferenceToFDFDoc](#)  
[FPDObjectCloneRefToFDFDoc](#)  
[FPDReferenceNew2](#)  
[FPDReferenceSetRefToFDFDoc](#)  
[FPDFDFDocAddIndirectObject](#)  
[FPDFDFDocClearModified](#)  
[FPDFDFDocDeleteIndirectObject](#)  
[FPDFDFDocDestroy](#)  
[FPDFDFDocExportAnnotToPDFPage](#)  
[FPDFDFDocGetAnnotCount](#)  
[FPDFDFDocGetAnnotDict](#)  
[FPDFDFDocGetIndirectObject](#)  
[FPDFDFDocGetIndirectType](#)  
[FPDFDFDocGetLastobjNum](#)  
[FPDFDFDocGetNextAssoc](#)  
[FPDFDFDocGetRoot](#)  
[FPDFDFDocGetPosition](#)  
[FPDFDFDocGetWin32Path](#)  
[FPDFDFDocImportExternalObject](#)  
[FPDFDFDocImportIndirectObject](#)  
[FPDFDFDocImportPDFAnnot](#)  
[FPDFDFDocInsertIndirectObject](#)  
[FPDFDFDocIsModified](#)  
[FPDFDFDocReleaseIndirectObject](#)  
[FPDFDFDocReloadFileStreams](#)  
[FPDFDFDocSetPDFPath](#)  
[FPDFDFDocWriteBuf](#)  
[FPDFDFDocWriteFile](#)

## Functions

### Functions summary

#### **FPDFDFDocAddIndirectObject**

Adds an object to indirect object list. The new object number is returned.

#### **FPDFDFDocClearModified**

Clears the modified flag.

#### **FPDFDFDocDeleteIndirectObject**

Deletes an memory document. Use this function with caution!

#### **FPDFDFDocDestroy**

Destroys The FDF document.

#### **FPDFDFDocExportAnnotToPDFPage**

Export an annotation object loaded from a FDF document into a PDF page.

#### **FPDFDFDocGetAnnotCount**

Count annotations in a FDF document by specific filter.

#### **FPDFDFDocGetAnnotDict**

Get an annotation dictionary from FDF document by specific filter and index.

#### **FPDFDFDocGetAnnotPageIndex**

Get the zero-based index of PDF page on which the original PDF annotation locates.

#### **FPDFDFDocGetIndirectObject**

Loads an indirect object by an object number.

#### **FPDFDFDocGetIndirectType**

Gets type of an memory document, without loading the object content.

#### **FPDFDFDocGetLastobjNum**

Gets last assigned indirect object number.

#### **FPDFDFDocGetNextAssoc**

Accesses the indirect object of current position, and move the position to next.

#### **FPDFDFDocGetRoot**

Gets root dictionary of the FDF document. All other data can be fetched from this root.

#### **FPDFDFDocGetPosition**

Gets the start position of the indirect objects.

#### **FPDFDFDocGetWin32Path**

Gets file path of attached PDF document, if any. Different platform has different format.

#### **FPDFDFDocImportExternalObject**

Imports an object from external object collection as a new memory document. If the external object refers to other external indirect objects, those indirect objects are also imported. The mapping from old object number to new object number is updated during the import process.

#### **FPDFDFDocImportIndirectObject**

Imports an object from its binary format. This is used when the PDF is fetched in "Progressive Downloading" fashion. After this function call, the data buffer can be destroyed. This object must not be encrypted.

#### **FPDFDFDocImportPDFAnnot**

Import an annotation from PDF document.

#### **FPDFDFDocInsertIndirectObject**

Inserts an indirect object with specified object number. This is used when the PDF is fetched in "Progressive Downloading" fashion, or parsing FDF. If an indirect object with the same object number exists, the previous one will be destroyed.

#### **FPDFDFDocIsModified**

Checks if any of the indirect objects are modified, since loading from parser, or last ClearModified.

**FPDFDFDocNew**

Creates a new empty FDF document.

**FPDFDFDocOpenFromFile**

Loads a FDF document from a file.

**FPDFDFDocPareMemory**

Loads a FDF document from a memory block.

**FPDFDFDocReleaseIndirectObject**

Sometimes, for saving memory space, we can release a loaded memory document. However, use this with caution because the object pointer will become invalid.

**FPDFDFDocReloadFileStreams**

Reload all file based stream when we do reparsing.

**FPDFDFDocSetPDFPath**

Set the path of PDF document to FDF document.

**FPDFDFDocWriteBuf**

Writes the content of this FDF document to a memory block.

**FPDFDFDocWriteFile**

Writes the content of this FDF document to a file.

**Functions detail****FPDFDFDocAddIndirectObject****Syntax**

```
FS_DWORD FPDFDFDocAddIndirectObject (
    FDF Document doc,
    FPD Object obj
);
```

**Description**

Adds an object to indirect object list. The new object number is returned.

**Parameter**

doc	[In] The input memory document.
-----	---------------------------------

obj	[In] The input object.
-----	------------------------

**Return**

The new object number.

**Head file reference**

fpd\_docTempl.h: 6151

**FPDFDFDocClearModified****Syntax**

```
void FPDFDFDocClearModified (
    FDF Document doc
);
```

**Description**

Clears the modified flag.

**Parameter**

---

doc	[In] The input memory document.
-----	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 6271

**FPDFDFDocDeleteIndirectObject****Syntax**

```
void FPDFDFDocDeleteIndirectObject (
    FDF Document doc,
    FS\_DWORD objNum
);
```

**Description**

Deletes an memory document. Use this function with caution!

**Parameter**

---

doc	[In] The input memory document.
-----	---------------------------------

---

objNum	[In] The object number to delete.
--------	-----------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 6172

**FPDFDFDocDestroy****Syntax**

```
void FPDFDFDocDestroy (
    FDF Document doc
);
```

**Description**

Destroys The FDF document.

**Parameter**

---

doc	[In] The input FDF document.
-----	------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 6093

**FPDFDFDocExportAnnotToPDFPage****Syntax**

```
FS_BOOL FPDFDFDocExportAnnotToPDFPage (
    FDF Document doc,
    FPD Object fdfAnnotDict,
    FPD Document pdfDoc,
    FPD Page pdfPage,
    FS PtrArray* arrAnnotDict
);
```

**Description**

Export an annotation object loaded from a FDF document into a PDF page.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

---

fdfAnnotDict	[In] The FDF annotation dictionary.
--------------	-------------------------------------

---

---

pdfDoc	[In] The PDF document to which the FDF annotation is exported.
--------	--

---

---

pdfPage	[In] The PDF page to which the FDF annotation is exported.
---------	--

---

---

arrAnnotDict	[Out] The pointer array to receive generated PDF annotation dictionaries.
--------------	---

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 6325

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FPDFDFDocGetAnnotCount****Syntax**

```
FS_INT32 FPDFDFDocGetAnnotCount (
    FDF\_Document doc,
    FS\_LPCSTR szFilter
);
```

**Description**

Count annotations in a FDF document by specific filter.

**Parameter**

doc	[In] The input document.
szFilter	[In] Pointer to a filter string. It can be NULL, otherwise it must be like "Text" or "Text, Link, Circle".

**Return**

The count of annotations.

**Head file reference**

fpd\_docTempl.h: 6290

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FPDFDFDocGetAnnotDict****Syntax**

```
FPD_Object FPDFDFDocGetAnnotDict (
    FDF\_Document doc,
    FS\_LPCSTR szFilter,
    FS\_INT32 nIndex
);
```

**Description**

Get an annotation dictionary from FDF document by specific filter and index.

**Parameter**

doc	[In] The input document.
-----	--------------------------

---

szFilter	[In] Pointer to a filter string. It can be NULL, otherwise it must be like "Text" or "Text, Link, Circle".
----------	--

---

nIndex	[In] A zero-based annotation index based on the annotations which are specified by filter.
--------	--

---

**Return**

The FDF annotation dictionary.

**Head file reference**

fpd\_docTempl.h: 6302

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FPDFDFDocGetAnnotPageIndex****Syntax**

```
FS_INT32 FPDFDFDocGetAnnotPageIndex (
    FPD Object fdfAnnotDict
);
```

**Description**

Get the zero-based index of PDF page on which the original PDF annotation locates.

**Parameter**


---

fdfAnnotDict	[In] The FDF annotation dictionary.
--------------	-------------------------------------

---

**Return**

The zero-based index of PDF page.

**Head file reference**

fpd\_docTempl.h: 6315

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FPDFDFDocGetIndirectObject****Syntax**

```
FPD Object FPDFDFDocGetIndirectObject (
    FDF Document doc,
    FS DWORD objNum
);
```

**Description**

Loads an indirect object by an object number.

**Parameter**

---

doc	[In] The input memory document.
-----	---------------------------------

---

objNum	[In] The input object number.
--------	-------------------------------

---

**Return**

An memory document.

**Head file reference**

fpd\_docTempl.h: 6131

**FPDFDFDocGetIndirectType****Syntax**

```
FS_INT32 FPDFDFDocGetIndirectType (
    FDF Document doc,
    FS DWORD objNum
);
```

**Description**

Gets type of an memory document, without loading the object content.

**Parameter**

---

doc	[In] The input memory document.
-----	---------------------------------

---

objNum	[In] The input object number.
--------	-------------------------------

---

**Return**

The type of the memory document.

**Head file reference**

fpd\_docTempl.h: 6141

**FPDFDFDocGetLastobjNum****Syntax**

```
FS_DWORD FPDFDFDocGetLastobjNum (
    FDF Document doc
);
```

**Description**

---

Gets last assigned indirect object number.

#### Parameter

---

doc	[In] The input memory document.
-----	---------------------------------

---

#### Return

Last assigned indirect object number.

#### Head file reference

fpd\_docTempl.h: 6223

### FPDFDFDocGetNextAssoc

#### Syntax

```
void FPDFDFDocGetNextAssoc (
    FDF Document doc,
    FS\_POSITION* outPos,
    FS\_DWORD* outObjNum,
    FPD Object* outObject
);
```

#### Description

Accesses the indirect object of current position, and move the position to next.

#### Parameter

---

doc	[In] The input memory document.
-----	---------------------------------

---

outPos	[In/Out] Input current position and receive the next position.
--------	--

---

outObjNum	[Out] It receives the current object number.
-----------	--

---

outObject	[Out] It receives the current indirect object pointer.
-----------	--

---

#### Return

void

#### Head file reference

fpd\_docTempl.h: 6250

### FPDFDFDocGetRoot

#### Syntax

```
FPD_Object FPDFDFDocGetRoot (
```

```
FDF_Document doc  
);
```

**Description**

Gets root dictionary of the FDF document. All other data can be fetched from this root.

**Parameter**

---

doc	[In] The input FDF document.
-----	------------------------------

---

**Return**

Root dictionary of the FDF document.

**Head file reference**

fpd\_docTempl.h: 6112

**FPDFDFDocGetPosition****Syntax**

```
FS_POSITION FPDFDFDocGetPosition (  
    FDF_Document doc  
);
```

**Description**

Gets the start position of the indirect objects.

**Parameter**

---

doc	[In] The input memory document.
-----	---------------------------------

---

**Return**

The start position of the indirect objects.

**Head file reference**

fpd\_docTempl.h: 6241

**FPDFDFDocGetWin32Path****Syntax**

```
void FPDFDFDocGetWin32Path (  
    FDF_Document doc,  
    FS_WideString* outPath  
);
```

**Description**

Gets file path of attached PDF document, if any. Different platform has different format.

**Parameter**

---

doc	[In] The input FDF document.
-----	------------------------------

---

outPath	[Out] The file path of attached PDF document, if any.
---------	---

---

**Return**

Root dictionary of the FDF document.

**Head file reference**

fpd\_docTempl.h: 6121

**FPDFDFDocImportExternalObject****Syntax**

```
FPD_Object FPDFDFDocImportExternalObject (
    FDF Document doc,
    FPD Object obj,
    FS MapPtrToPtr mapping
);
```

**Description**

Imports an object from external object collection as a new memory document. If the external object refers to other external indirect objects, those indirect objects are also imported. The mapping from old object number to new object number is updated during the import process.

**Parameter**

---

doc	[In] The input memory document.
-----	---------------------------------

---

obj	[In] The object in external object collection.
-----	--

---

mapping	[Out] It updates the mapping from old object number to new object number.
---------	---

---

**Return**

A new memory document.

**Head file reference**

fpd\_docTempl.h: 6196

**FPDFDFDocImportIndirectObject****Syntax**

```
FPD_Object FPDFDFDocImportIndirectObject (
    FDF Document doc,
    FS\_LPCBYTE pBuffer,
    FS\_DWORD size
);
```

**Description**

Imports an object from its binary format. This is used when the PDF is fetched in "Progressive Downloading" fashion. After this function call, the data buffer can be destroyed. This object must not be encrypted.

**Parameter**

---

doc	[In] The input memory document.
pBuffer	[In] The binary data for the memory document. It must be not encrypted.
size	[In] The size in bytes of the binary data.

---

**Return**

An object.

**Head file reference**

fpd\_docTempl.h: 6182

**FPDFDFDocImportPDFAnnot****Syntax**

```
FS_BOOL FPDFDFDocImportPDFAnnot (
    FDF Document doc,
    FPD Object pdfAnnotDict,
    FS\_INT32 nPageIndex
);
```

**Description**

Import an annotation from PDF document.

**Parameter**

---

doc	[In] The input document.
pdfAnnotDict	[In] The PDF annotation dictionary to be imported.

---

---

nPageIndex	[In] The zero-based index of PDF page on which the annotation locates.
------------	--

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 6339

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FPDFDFDocInsertIndirectObject****Syntax**

```
void FPDFDFDocInsertIndirectObject (
    FDF Document doc,
    FS\_DWORD objNum,
    FPD Object obj
);
```

**Description**

Inserts an indirect object with specified object number. This is used when the PDF is fetched in "Progressive Downloading" fashion, or parsing FDF. If an indirect object with the same object number exists, the previous one will be destroyed.

**Parameter**


---

doc	[In] The input memory document.
-----	---------------------------------

---



---

objNum	[In] The new object number of the indirect object to insert.
--------	--

---



---

obj	[In] The indirect object to insert.
-----	-------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 6210

**FPDFDFDocIsModified****Syntax**

```
FS_BOOL FPDFDFDocIsModified (
    FDF Document doc
);
```

**Description**

Checks if any of the indirect objects are modified, since loading from parser, or last ClearModified.

**Parameter**

---

doc	[In] The input memory document.
-----	---------------------------------

---

**Return**

TRUE for the indirect objects are modified, otherwise not.

**Head file reference**

fpd\_docTempl.h: 6262

**FPDFDFDocNew****Syntax**

```
FDF_Document FPDFDFDocNew (void );
```

**Description**

Creates a new empty FDF document.

**Return**

A new empty FDF document.

**Head file reference**

fpd\_docTempl.h: 6066

**FPDFDFDocOpenFromFile****Syntax**

```
FDF_Document FPDFDFDocOpenFromFile (
    FS_LPCSTR strFilePath
);
```

**Description**

Loads a FDF document from a file.

**Parameter**

---

strFilePath	[In] The input file full path.
-------------	--------------------------------

---

**Return**

The FDF document.

**Head file reference**

fpd\_docTempl.h: 6074

**FPDFDFDocPareMemory****Syntax**

```
FDF_Document FPDFDFDocPareMemory (
    FS_LPCBYTE memoryBlock,
    FS_DWORD size
);
```

**Description**

Loads a FDF document from a memory block.

**Parameter**

---

memoryBlock	[In] The input memory block.
-------------	------------------------------

---

size	[In] The size in bytes of the block.
------	--------------------------------------

---

**Return**

The FDF document.

**Head file reference**

fpd\_docTempl.h: 6083

**FPDFDFDocReleaseIndirectObject****Syntax**

```
void FPDFDFDocReleaseIndirectObject (
    FDF_Document doc,
    FS_DWORD objNum
);
```

**Description**

Sometimes, for saving memory space, we can release a loaded memory document. However, use this with caution because the object pointer will become invalid.

**Parameter**

---

doc	[In] The input memory document.
-----	---------------------------------

---

objNum	[In] The object number to release.
--------	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 6161

**FPDFDFDocReloadFileStreams****Syntax**

```
void FPDFDFDocReloadFileStreams (
    FDF Document doc
);
```

**Description**

Reload all file based stream when we do reparsing.

**Parameter**

---

doc	[In] The input memory document.
-----	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 6232

**FPDFDFDocSetPDFPath****Syntax**

```
void FPDFDFDocSetPDFPath (
    FDF Document doc,
    FS LPCWSTR wszPDFPath
);
```

**Description**

Set the path of PDF document to FDF document.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

wszPDFPath	[In] The PDF path.
------------	--------------------

---

**Return****Head file reference**

fpd\_docTempl.h: 6351

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FPDFDFDocWriteBuf****Syntax**

```
FS_BOOL FPDFDFDocWriteBuf (
    FDF\_Document doc,
    FS\_ByteString* outBuf
);
```

**Description**

Writes the content of this FDF document to a memory block.

**Parameter**

---

doc	[In] The input document.
outBuf	[Out] It will receive the content of this PDF document.

---

**Return**

[TRUE](#) means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 6280

**FPDFDFDocWriteFile****Syntax**

```
FS_BOOL FPDFDFDocWriteFile (
    FDF\_Document doc,
    FS\_LPCSTR strFilePath
);
```

**Description**

Writes the content of this FDF document to a file.

**Parameter**

---

doc	[In] The input FDF document.
strFilePath	[In] The file full path to write to.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**  
fpd\_docTempl.h: 6102

## **FDRM\_PDFSecurityHandler**

[Description](#)

## **FOFD\_ACTION**

[Description](#)

## **FOFD\_ActionArea**

[Description](#)

## **FOFD\_ActionGoto**

[Description](#)

## **FOFD\_ActionGotoA**

[Description](#)

## **FOFD\_ActionRegion**

[Description](#)

## **FOFD\_ACTIONS**

[Description](#)

## **FOFD\_ANNOT**

[Description](#)

## **FOFD\_Bookmark**

[Description](#)

## **FOFD\_Bookmarks**

[Description](#)

## **FOFD\_CREATOR**

**Description**

## **FOFD\_CRYPTODICT**

**Description**

## **FOFD\_CryptoHandler**

**Description**

## **FOFD\_DEST**

**Description**

## **FOFD\_DIBAttribute**

**Description**

## **FOFD\_Document**

**Description**

The underlying OFD representation of a document. you can use this object to write data to the document.

## **FOFD\_DriverDevice**

**Description**

## **FOFD\_FILEPACKAGE**

**Description**

## **FOFD\_FileStream**

**Description**

## **FOFD\_OBJECT**

**Description**

The underlying OFD representation of a object. you can use this object to read data from the object.

## FOFD\_OUTLINE

### Description

## FOFD\_PAGE

### Description

The underlying OFD representation of a page. you can use this object to read data from the page.

## FOFD\_PARSER

### Description

## FOFD\_Path

### Description

## FOFD\_PauseHandler

### [Return from Used by](#)

### Description

An object preparing a simple pause instance.

### Returned from

### [FOFDPauseHandlerCreate](#)

### Used by

### [FOFDPauseHandlerDestroy](#)

## Functions

### Functions summary

#### [FOFDPauseHandlerCreate](#)

Creates the pause handler.

#### [FOFDPauseHandlerDestroy](#)

Destroys the pause handler.

### Functions detail

## FOFDPauseHandlerCreate

### Syntax

```
FOFD_PauseHandler FOFDPauseHandlerCreate (
    OFD\_Pause pause
);
```

### Description

Creates the pause handler.

### Parameter

---

pause	[In] The input pause handler structure.
-------	---

---

### Return

The newly created pause handler.

### Head file reference

fofd\_basicTempl.h: 192

## FOFDPauseHandlerDestroy

### Syntax

```
void FOFDPauseHandlerDestroy (
    FOFD\_PauseHandler pauseHandler
);
```

### Description

Destroys the pause handler.

### Parameter

---

pauseHandler	[In] The input pause handler to be destroyed.
--------------	---

---

### Return

void

### Head file reference

fofd\_basicTempl.h: 201

## FOFD\_PERMISSIONS

### Description

## FOFD\_ProgressiveRenderer

**Description**

## **FOFD\_RenderContext**

**Description**

## **FOFD\_RenderDevice**

**Description**

## **FOFD\_RenderOptions**

**Description**

## **FOFD\_SecurityHandler**

**Description**

## **FOFD\_SIGNATURE**

**Description**

The underlying OFD representation of a Signature. you can use this object to read data from the Signature.

## **FOFD\_VPREFERENCES**

**Description**

## **FOFD\_WRITEPAGE**

**Description**

The underlying OFD representation of a page. you can use this object to write data to the page.

## **FOFD\_WRITESIGNATURE**

**Description**

The underlying OFD representation of a Signature. you can use this object to write data to the Signature.

## **FPD\_AAction**

## [\*\*Return from Used by\*\*](#)

### Description

Additional-action extends a set of events that can trigger the execution of actions. It is specified in /AA entry in a annotation, page object, interactive form or document catalog. See [FPDAActionNew](#) , [FPDAActionDestroy](#) , [FPDFFormFieldGetAdditionalAction](#) , [FPDFFormControlGetAdditionalAction](#) .

### Returned from

#### [FPDAActionNew](#)

### Used by

[FPDFFormControlGetAdditionalAction](#)  
[FPDFFormFieldGetAdditionalAction](#)  
[FPDFFormFieldSetAdditionalAction](#)  
[FPDAActionActionExist](#)  
[FPDAActionDestroy](#)  
[FPDAActionGetAction](#)  
[FPDAActionGetDictionary](#)  
[FPDAActionGetNextAction](#)  
[FPDAActionGetStartPos](#)  
[FPDAActionRemoveAction](#)  
[FPDAActionSetAction](#)

### Enumerations

#### Enumerations summary

##### [FPD\\_A ActionType](#)

Additional-action type enumeration.

#### Enumerations detail

##### [FPD\\_A ActionType](#)

###### Syntax

```
enum FPD_A ActionType{  
    CursorEnter,  
    CursorExit,  
    ButtonDown,  
    ButtonUp,  
    GetInputFocus,  
    LoseInputFocus,  
    PageOpen,  
    PageClose,  
    PageVisible,  
    PageInvisible,  
    ClosePage,  
    Format,  
    Validate,  
    Calculate,  
    SaveDocument,
```

```
    DocumentSaved,  
    PrintDocument,  
    DocumentPrinted  
};
```

**Description**

Additional-action type enumeration.

**Head file reference**

fpd\_docExpT.h: 620

**CursorEnter**

E, cursor enters annotation, arbitrary action.

**CursorExit**

X, cursor exits annotation, arbitrary action.

**ButtonDown**

D, mouse button is pressed, arbitrary action.

**ButtonUp**

U, mouse button is released, arbitrary action.

**GetInputFocus**

Fo, annotation get input focus, arbitrary action.

**LoseInputFocus**

Bl, annotation loses input focus, arbitrary action.

**PageOpen**

PO, page is opened, executed after O, arbitrary action.

**PageClose**

PC, page is closed, executed before C, arbitrary action.

**PageVisible**

PV, page becomes visible, arbitrary action.

**PageInvisible**

PI, page is no longer visible, arbitrary action.

**ClosePage**

C, page is closed, arbitrary action.

**Format**

F, field is to be formatted, JavaScript action.

**Validate**

V, field is to be validated, JavaScript action.

**Calculate**

C, recalculate value, JavaScript action.

**SaveDocument**

WS, before saving document, JavaScript action.

**DocumentSaved**

DS, after saving document, JavaScript action.

**PrintDocument**

WP, before printing document, JavaScript action.

**DocumentPrinted**

DP, after printing document, JavaScript action.

## Functions

### Functions summary

**[FPDAActionActionExist](#)**

Checks whether the specified additional-action type exists or not.

**[FPDAActionDestroy](#)**

Destroys PDF additional-action.

**[FPDAActionGetAction](#)**

Gets an action from the additional-action with specified additional-action type.

**[FPDAActionGetDictionary](#)**

Gets the additional-action dictionary.

**[FPDAActionGetNextAction](#)**

Gets the current action and move the position to next position.

**[FPDAActionGetStartPos](#)**

Gets the start position of action list.

**[FPDAActionNew](#)**

Creates PDF additional-action from a PDF dictionary.

**[FPDAActionRemoveAction](#)**

Removes a additional-action.

**[FPDAActionSetAction](#)**

Sets a additional-action of specified type.

### Functions detail

**FPDAActionActionExist****Syntax**

```
FS_BOOL FPDAActionActionExist (
    FPD_AAction action,
    FPD_AActionType eType
);
```

**Description**

Checks whether the specified additional-action type exists or not.

**Parameter**

---

aaction	[In] PDF additional-action
---------	----------------------------

---

eType	[In] The input additional-action type.
-------	--

---

**Return**

Non-zero means exist, otherwise not exist.

**Head file reference**

fpd\_docTempl.h: 2275

**FPDAActionDestroy****Syntax**

```
void FPDAActionDestroy (
    FPD\_AAction aaction
);
```

**Description**

Destroys PDF additional-action.

**Parameter**

---

aaction	[In] PDF additional-action
---------	----------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2266

**FPDAActionGetAction****Syntax**

```
void FPDAActionGetAction (
    FPD\_AAction aaction,
    FPD\_AActionType eType,
    FPD\_Action* outAction
);
```

**Description**

Gets an action from the additional-action with specified additional-action type.

**Parameter**

---

aaction	[In] PDF additional-action
eType	[In] The input additional-action type.
outAction	[Out] An action.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2285

**FPDAACTIONGETDICTIONARY****Syntax**

```
FPD_Object FPDAACTIONGETDICTIONARY (
    FPD\_AAction action
);
```

**Description**

Gets the additional-action dictionary.

**Parameter**


---

aaction	[In] PDF additional-action
---------	----------------------------

---

**Return**

The additional-action dictionary.

**Head file reference**

fpd\_docTempl.h: 2339

**Since**[SDK\\_LATEEST\\_VERSION > 7.2.2](#)**FPDAACTIONGETNEXTACTION****Syntax**

```
void FPDAACTIONGETNEXTACTION (
    FPD\_AAction action,
    FS\_POSITION* outPos,
    FPD\_AActionType* outType,
    FPD\_Action* outAction
);
```

**Description**

---

Gets the current action and move the position to next position.

**Parameter**

aaction	[In] PDF additional-action
outPos	[In/Out] Input the current position and receive the next position.
outType	[Out] Receive the additional-action type.
outAction	[Out] Receive the current action.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2327

**FPDAActionGetStartPos****Syntax**

```
FS_POSITION FPDAActionGetStartPos (
    FPD\_AAction action
);
```

**Description**

Gets the start position of action list.

**Parameter**

aaction	[In] PDF additional-action
---------	----------------------------

**Return**

The start position of action list.

**Head file reference**

fpd\_docTempl.h: 2318

**FPDAActionNew****Syntax**

```
FPD_AAction FPDAActionNew (
    FPD\_Object dict
);
```

**Description**

Creates PDF additional-action from a PDF dictionary.

**Parameter**

---

dict	[In] The input PDF dictionary.
------	--------------------------------

---

**Return**

PDF additional-action.

**Head file reference**

fpd\_docTempl.h: 2257

**FPDAActionRemoveAction****Syntax**

```
void FPDAActionRemoveAction (
    FPD\_AAction action,
    FPD\_AActionType eType
);
```

**Description**

Removes a additional-action.

**Parameter**

---

aaction	[In] PDF additional-action
---------	----------------------------

---

eType	[In] The additional-action type to be removed.
-------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2308

**FPDAActionSetAction****Syntax**

```
void FPDAActionSetAction (
    FPD\_AAction action,
    FPD\_Document doc,
    FPD\_AActionType eType,
    const FPD\_Action action
);
```

**Description**

Sets a additional-action of specified type.

**Parameter**

aaction	[In] PDF additional-action
doc	[In] The PDF document.
eType	[In] The additional-action type.
action	[In] The input action.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2296

## FPD\_Action

### [Return from Used by](#)

#### Description

Actions are what happens when a user clicks on a link or bookmark. In addition, the Foxit Reader allows a document to have an action that is executed automatically when the document is opened. See [FPDACTIONNew](#) , [FPDACTIONNew2](#) , [FPDACTIONNew3](#) , [FPDACTIONDestroy](#) , [FPDAACTIONGetAction](#) , [FPDAACTIONGetNextAction](#) , [FPDDocJSActionsGetJSAction2](#) , [FPDLinkGetAction](#) , [FPDBookmarkGetAction](#) .

#### Returned from

[FPDACTIONFieldsGetAction](#)  
[FPDACTIONFieldsNew](#)  
[FPDACTIONNew](#)  
[FPDACTIONNew2](#)  
[FPDACTIONNew3](#)

#### Used by

[FRDocViewProcAction](#)  
[FPDAACTIONGetAction](#)  
[FPDAACTIONGetNextAction](#)  
[FPDACTIONFieldsDestroy](#)  
[FPDACTIONFieldsGetAction](#)

[FPDACTIONFIELDSGETALLFIELDS](#)  
[FPDACTIONFIELDSGETFIELD](#)  
[FPDACTIONFIELDSGETFIELDSCOUNT](#)  
[FPDACTIONFIELDSINSERTFIELD](#)  
[FPDACTIONFIELDSNEW](#)  
[FPDACTIONFIELDSREMOVEFIELD](#)  
[FPDBOOKMARKGETACTION](#)  
[FPDDOCJSActionsGetJSAction](#)  
[FPDDOCJSActionsGetJSAction2](#)  
[FPDDOCJSActionsSetJSAction](#)  
[FPDFFORMCONTROLGETACTION](#)  
[FPDLINKGETACTION](#)  
[FPDACTIONCOUNTOCGSTATES](#)  
[FPDACTIONCOUNTRENDITIONS](#)  
[FPDACTIONDESTROY](#)  
[FPDACTIONGETANNOT](#)  
[FPDACTIONGETDEST](#)  
[FPDACTIONGETDICT](#)  
[FPDACTIONGETFILEPATH](#)  
[FPDACTIONGETFLAGS](#)  
[FPDACTIONGETHIDESTATUS](#)  
[FPDACTIONGETJAVASCRIPT](#)  
[FPDACTIONGETMOUSEMAP](#)  
[FPDACTIONGETNAMEACTION](#)  
[FPDACTIONGETNEWWINDOW](#)  
[FPDACTIONGETOCGSTATES](#)  
[FPDACTIONGETOPERATIONTYPE](#)  
[FPDACTIONGETRENDITION](#)  
[FPDACTIONGETSOUNDSTREAM](#)  
[FPDACTIONGETSUBACTION](#)  
[FPDACTIONGETSUBACTIONSCOUNT](#)  
[FPDACTIONGETTYPE](#)  
[FPDACTIONGETTYPENAME](#)  
[FPDACTIONGETURI](#)  
[FPDACTIONGETVOLUME](#)  
[FPDACTIONGETWIDGETS](#)  
[FPDACTIONGETWINPARAM](#)  
[FPDACTIONINSERTOCGSTATES](#)  
[FPDACTIONINSERTRENDITION](#)  
[FPDACTIONINSERTSUBACTION](#)  
[FPDACTIONISMIXPLAY](#)  
[FPDACTIONISREPEAT](#)  
[FPDACTIONISSTATEPRESERVED](#)  
[FPDACTIONISSYNCHRONOUS](#)  
[FPDACTIONNEW2](#)  
[FPDACTIONREMOVEALLSUBACTIONS](#)  
[FPDACTIONREMOVEOCGSTATES](#)  
[FPDACTIONREMOVERENDITION](#)  
[FPDACTIONREMOVESUBACTION](#)  
[FPDACTIONREPLACEOCGSTATES](#)  
[FPDACTIONSETANNOT](#)  
[FPDACTIONSETDEST](#)  
[FPDACTIONSETFILEPATH](#)  
[FPDACTIONSETFLAGS](#)  
[FPDACTIONSETHIDESTATUS](#)  
[FPDACTIONSETJAVASCRIPT](#)  
[FPDACTIONSETJAVASCRIPTW](#)

[FPDActionSetMouseMap](#)  
[FPDActionSetNameAction](#)  
[FPDActionSetNewWindow](#)  
[FPDActionSetOperationType](#)  
[FPDActionSetStatePreserved](#)  
[FPDActionSetURI](#)  
[FPDActionSetWinParam](#)

## Definitions

### Definitions summary

#### [FPD\\_NAMED\\_FIRSTPAGE](#)

Go to the first page of the document.

#### [FPD\\_NAMED\\_LASTPAGE](#)

Go to the last page of the document.

#### [FPD\\_NAMED\\_NEXTPAGE](#)

Go to the next page of the document.

#### [FPD\\_NAMED\\_PREVPAGE](#)

Go to the previous page of the document.

#### [FPD\\_JS\\_MAXLENGTH](#)

macro for JavaScript string, maximum length.

### Definitions detail

#### FPD\_NAMED\_FIRSTPAGE

##### **Syntax**

```
#define FPD_NAMED_FIRSTPAGE 3
```

##### **Description**

Go to the first page of the document.

##### **Group**

[FPDActionTypeNamed](#)

##### **Head file reference**

fpd\_docExpT.h: 530

#### FPD\_NAMED\_LASTPAGE

##### **Syntax**

```
#define FPD_NAMED_LASTPAGE 4
```

##### **Description**

Go to the last page of the document.

##### **Group**

[FPDActionTypeNamed](#)

**Head file reference**

fpd\_docExpT.h: 532

**FPD\_NAMED\_NEXTPAGE****Syntax**

#define FPD\_NAMED\_NEXTPAGE 1

**Description**

Go to the next page of the document.

**Group**[FPDACTIONTYPENAMED](#)**Head file reference**

fpd\_docExpT.h: 526

**FPD\_NAMED\_PREVPAGE****Syntax**

#define FPD\_NAMED\_PREVPAGE 2

**Description**

Go to the previous page of the document.

**Group**[FPDACTIONTYPENAMED](#)**Head file reference**

fpd\_docExpT.h: 528

**FPD\_JS\_MAXLENGTH****Syntax**

#define FPD\_JS\_MAXLENGTH 64

**Description**

macro for JavaScript string, maximum length.

**Group**[FPDJavaScriptString](#)**Head file reference**

fpd\_docExpT.h: 544

## Enumerations

### Enumerations summary

#### **FPD\_ActionType**

PDF Action Type Enumeration. See [FPDActionNew2](#) .

### Enumerations detail

#### **FPD\_ActionType**

##### **Syntax**

```
enum FPD_ActionType{
    UnknownType,
    GoTo,
    GoToR,
    GoToE,
    Launch,
    Thread,
    URI,
    Sound,
    Movie,
    Hide,
    Named,
    SubmitForm,
    ResetForm,
    ImportData,
    JavaScript,
    SetOCGState,
    Rendition,
    Trans,
    GoTo3DView
};
```

##### **Description**

PDF Action Type Enumeration. See [FPDActionNew2](#) .

##### **Head file reference**

fpd\_docExpT.h: 554

##### **UnknownType**

Unknown action type.

##### **GoTo**

Go-To Actions.

##### **GoToR**

Remote Go-To Actions.

##### **GoToE**

Embedded Go-To Actions.

##### **Launch**

Launch Actions.

**Thread**

Thread Actions.

**URI**

URI Actions.

**Sound**

Sound Actions.

**Movie**

Movie Actions.

**Hide**

Hide Actions.

**Named**

Named Actions.

**SubmitForm**

Submit-Form Actions.

**ResetForm**

Reset-Form Actions.

**ImportData**

Import-Data Actions.

**JavaScript**

JavaScript Actions.

**SetOCGState**

Set-OCG-State Actions.

**Rendition**

Rendition Actions.

**Trans**

Transition Actions.

**GoTo3DView**

Go-To-3D-View Actions.

## Functions

### Functions summary

**FPDACTIONCOUNTOCGSTATES**

Gets the count of OCG states.

**FPDACTIONCOUNTRENDITIONS**

Gets the count of renditions.

**FPDACTIONDESTROY**

Destroys a PDF action.

**FPDACTIONGETANNOT**

Gets the annotation dictionary.

**FPDACTIONGETDEST**

Gets the destination.

**FPDACTIONGETDICT**

Gets the PDF action dictionary.

**FPDACTIONGETFILEPATH**

Gets the file full path.

**FPDACTIONGETFLAGS**

Gets the flags for action type SubmitForm, ResetForm.

**FPDACTIONGETHIDESTATUS**

Gets the hide status of a PDF action.

**FPDACTIONGETJAVASCRIPT**

Gets the javascript script to be executed.

**FPDACTIONGETMOUSEMAP**

Gets the flag which indicates whether to track the mouse position when the URI is resolved.

**FPDACTIONGETNAMEACTION**

Gets the name of named action.

**FPDACTIONGETNEWWINDOW**

Checks whether to open the destination document in a new window or not.

**FPDACTIONGETOCGSTATES**

Gets OCG array with specified state.

**FPDACTIONGETOPERATIONTYPE**

Gets the operation type.

**FPDACTIONGETRENDITION**

Gets a rendition.

**FPDACTIONGETSOUNDSTREAM**

Gets the sound stream object.

**FPDACTIONGETSUBACTION**

Gets a sub-action.

**FPDACTIONGETSUBACTIONSCOUNT**

Gets the sub-action count.

**FPDACTIONGETTYPE**

Gets the type of the action.

**FPDACTIONGETTYPENAME**

Gets the type name of the action.

**FPDACTIONGETURI**

Gets the URI(uniform resource identifier) of the PDF action.

**FPDACTIONGETVOLUME**

Gets the volume of the sound. The volume at which to play the sound, in the range -1.0 to 1.0.

**FPDACTIONGETWIDGETS**

Gets the fields array.

**FPDACTIONGETWINPARAM**

Gets the windows launch param.

**FPDACTIONINSERTOCGSTATES**

Inserts a OCG state.

**FPDACTIONINSERTRENDITION**

Inserts a rendition.

**FPDACTIONINSERTSUBACTION**

Inserts a sub-action at specified position.

**FPDACTIONISMIXPLAY**

Checks whether to mix this sound with any other sound already playing.

**FPDACTIONISREPEAT**

Checks whether to repeat the sound indefinitely.

**FPDACTIONISSTATEPRESERVED**

Checks whether the radio-button state relationships between optional content groups apply or not.

**FPDACTIONISSYNCHRONOUS**

Checks whether to play the sound synchronously or asynchronously.

**FPDACTIONNEW**

Creates a PDF action from a PDF dictionary.

**FPDACTIONNEW2**

Creates a PDF action with specified action type.

**FPDACTIONNEW3**

Creates a PDF action with specified action type name.

**FPDACTIONREMOVEALLSUBACTIONS**

Removes all sub-actions.

**FPDACTIONREMOVEOCGSTATES**

Removes a OCG state.

**FPDACTIONREMOVErendition**

Removes a rendition.

**FPDACTIONREMOVESUBACTION**

Removes a sub-action.

**FPDACTIONREPLACEOCGSTATES**

Replaces a OCG state.

**FPDACTIONSETANNOT**

Sets the annotation dictionary.

**FPDACTIONSETDEST**

Sets the destination.

**FPDACTIONSETFILEPATH**

Sets the file full path.

**FPDACTIONSETFLAGS**

Sets the flags for action type SubmitForm, ResetForm.

**FPDACTIONSETHIDESTATUS**

Sets the hide status of a PDF action.

**FPDACTIONSETJAVASCRIPT**

Sets the javascript with a byte string.

**FPDACTIONSETJAVASCRIPTW**

Sets the javascript with a wide string.

**FPDACTIONSETMOUSEMAP**

Sets the mouse-position-tracking flag.

**FPDACTIONSETNAMEACTION**

Sets the name of named action.

**FPDACTIONSETNEWWINDOW**

Sets the new window flag.

**FPDACTIONSETOPERATIONTYPE**

Sets the operation type.

**FPDACTIONSETSTATEPRESERVED**

Sets the radio-button state relationships flag.

**FPDACTIONSETURI**

Sets the URI of the PDF action.

**FPDACTIONSETWINPARAM**

Sets the windows launch params.

## Functions detail

### FPDACTIONCOUNTOCGSTATES

**Syntax**

```
FS_INT32 FPDACTIONCOUNTOCGSTATES (
    FPD\_Action action
);
```

**Description**

Gets the count of OCG states.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

**Return**

The count of OCG states.

**Head file reference**

fpd\_docTempl.h: 2114

### FPDACTIONCOUNTRENDITIONS

**Syntax**

```
FS_INT32 FPDACTIONCOUNTRENDITIONS (
    FPD\_Action action
);
```

**Description**

Gets the count of renditions.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

**Return**

The count of renditions.

**Head file reference**

fpd\_docTempl.h: 1988

### FPDActionDestroy

#### Syntax

```
void FPDActionDestroy (
    FPD\_Action action
);
```

#### Description

Destroys a PDF action.

#### Parameter

---

action	[In] A PDF action.
--------	--------------------

---

#### Return

void

#### Head file reference

fpd\_docTempl.h: 1739

### FPDActionGetAnnot

#### Syntax

```
FPD_Object FPDActionGetAnnot (
    FPD\_Action action
);
```

#### Description

Gets the annotation dictionary.

#### Parameter

---

action	[In] A PDF action.
--------	--------------------

---

#### Return

The annotation dictionary.

#### Head file reference

fpd\_docTempl.h: 2030

### FPDActionGetDest

#### Syntax

```
void FPDActionGetDest (
```

```
FPD_Action action,  
FPD_Document doc,  
FPD_Dest* outDest  
);
```

**Description**

Gets the destination.

**Parameter**

action	[In] A PDF action.
doc	[In] The input PDF document.
outDest	[Out] A PDF destination.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1767

**FPDActionGetDict****Syntax**

```
FPD_Object FPDActionGetDict (  
    FPD_Action action  
) ;
```

**Description**

Gets the PDF action dictionary.

**Parameter**

action	[In] A PDF action.
--------	--------------------

**Return**

The PDF action dictionary.

**Head file reference**

fpd\_docTempl.h: 2240

**FPDActionGetFilePath****Syntax**

```
void FPDActionGetFilePath (
    FPD\_Action action,
    FS\_WideString* outName
);
```

**Description**

Gets the file full path.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

outName	[Out] It receives the file full path.
---------	---------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1788

**FPDActionGetFlags****Syntax**

```
FS_DWORD FPDActionGetFlags (
    FPD\_Action action
);
```

**Description**

Gets the flags for action type SubmitForm, ResetForm.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

**Return**

The flags for action type SubmitForm, ResetForm.

**Head file reference**

fpd\_docTempl.h: 1937

**FPDActionGetHideStatus****Syntax**

```
FS_BOOL FPDActionGetHideStatus (
    FPD\_Action action
);
```

**Description**

Gets the hide status of a PDF action.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

**Return**

The hide status of a PDF action.

**Head file reference**

fpd\_docTempl.h: 1898

**FPDActionGetJavaScript****Syntax**

```
void FPDActionGetJavaScript (
    FPD\_Action action,
    FS\_WideString* outJavaScript
);
```

**Description**

Gets the javascript script to be executed.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

outJavaScript	[Out] It receives the javascript script to be executed.
---------------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1956

**FPDActionGetMouseMap****Syntax**

```
FS_BOOL FPDActionGetMouseMap (
    FPD\_Action action
);
```

**Description**

Gets the flag which indicates whether to track the mouse position when the URI is resolved.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

**Return**

[TRUE](#) for tracking the mouse position when the URI is resolved.

**Head file reference**

fpd\_docTempl.h: 1869

**FPDActionGetNameAction****Syntax**

```
void FPDActionGetNameAction (
    FPD\_Action action,
    FS\_ByteString* outNamedAction
);
```

**Description**

Gets the name of named action.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

outNamedAction	[Out] It receives the name of named action.
----------------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1917

**FPDActionGetNewWindow****Syntax**

```
FS_BOOL FPDActionGetNewWindow (
    FPD\_Action action
);
```

**Description**

Checks whether to open the destination document in a new window or not.

**Parameter**


---

action	[In] A PDF action.
--------	--------------------

---

**Return**

[TRUE](#) for opening the destination document in a new window.

**Head file reference**

fpd\_docTempl.h: 1809

**FPDActionGetOCGStates****Syntax**

```
FS_BOOL FPDActionGetOCGStates (
    FPD\_Action action,
    FS\_INT32 iIndex,
    FS\_INT32* outState,
    FS\_PtrArray* outArrOcgs
);
```

**Description**

Gets OCG array with specified state.

**Parameter**


---

action	[In] A PDF action.
--------	--------------------

---



---

iIndex	[In] The zero-based OCG state index.
--------	--------------------------------------

---



---

outState	[Out] It receives the OCG state.
----------	----------------------------------

---



---

outArrOcgs	[Out] It receives the OCG state.
------------	----------------------------------

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 2123

**FPDActionGetOperationType****Syntax**

```
FS_INT32 FPDActionGetOperationType (
    FPD\_Action action
);
```

**Description**

Gets the operation type.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

**Return**

The operation type.

**Head file reference**

fpd\_docTempl.h: 2050

**FPDActionGetRendition****Syntax**

```
void FPDActionGetRendition (
    FPD\_Action action,
    FS\_INT32 index,
    FPD\_Rendition* outRendition
);
```

**Description**

Gets a rendition.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

---

index	[In] The zero-based rendition index.
-------	--------------------------------------

---

---

outRendition	[Out] It receives the rendition.
--------------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1997

**FPDActionGetSoundStream****Syntax**

```
FPD_Object FPDActionGetSoundStream (
    FPD\_Action action
);
```

**Description**

Gets the sound stream object.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

**Return**

The sound stream object.

**Head file reference**

fpd\_docTempl.h: 2069

**FPDActionGetSubAction****Syntax**

```
void FPDActionGetSubAction (
    FPD\_Action action,
    FS\_DWORD index,
);
```

**Description**

Gets a sub-action.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

---

index	[In] The zero-based sub-action index.
-------	---------------------------------------

---

[Out]

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2189

**FPDActionGetSubActionsCount****Syntax**

```
FS_DWORD FPDActionGetSubActionsCount (
    FPD\_Action action
);
```



**Description**

Gets the sub-action count.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

**Return**

The sub-action count.

**Head file reference**

fpd\_docTempl.h: 2189

**FPDActionGetType****Syntax**

```
FS_INT32 FPDActionGetType (
    FPD\_Action action
);
```

**Description**

Gets the type of the action.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

**Return**

The type of the action.

**Head file reference**

fpd\_docTempl.h: 1748

**FPDActionGetTypeName****Syntax**

```
void FPDActionGetTypeName (
    FPD\_Action action,
    FS\_ByteString* outTypeName
);
```

**Description**

Gets the type name of the action.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

outTypeName	[Out] It receives the type name of the action.
-------------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1748

**FPDActionGetURI****Syntax**

```
void FPDActionGetURI (
    FPD\_Action action,
    FPD\_Document doc,
    FS\_ByteString* outURL
);
```

**Description**

Gets the URI(uniform resource identifier) of the PDF action.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

doc	[In] The input PDF document.
-----	------------------------------

---

outURL	[Out] It receives the URI.
--------	----------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1848

**FPDActionGetVolume****Syntax**

```
FS_FLOAT FPDActionGetVolume (
    FPD\_Action action
);
```

**Description**

Gets the volume of the sound. The volume at which to play the sound, in the range -1.0 to 1.0.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

**Return**

The volume of the sound.

**Head file reference**

fpd\_docTempl.h: 2078

**FPDActionGetWidgets****Syntax**

```
void FPDActionGetWidgets (
    FPD\_Action action,
    FPD\_ActionFields* outWidgets
);
```

**Description**

Gets the fields array.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

---

outWidgets	[Out] It receives the action fields.
------------	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1888

**FPDActionGetWinParam****Syntax**

```
void FPDActionGetWinParam (
    FPD\_Action action,
    FPD\_LWinParam* outWinParam
);
```

**Description**

Gets the windows launch param.

**Parameter**

---

action	[In] A PDF action.
outWinParam	[Out] The windows launch params.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1828

**FPDActionInsertOCGStates****Syntax**

```
FS_INT32 FPDActionInsertOCGStates (
    FPD\_Action action,
    FPD\_Document doc,
    FS\_INT32 iIndex,
    FPD\_OCGState OCGState,
    const FS\_PtrArray arrOcgs
);
```

**Description**

Inserts a OCG state.

**Parameter**

---

action	[In] A PDF action.
doc	[In] The input PDF document.
iIndex	[In] The OCG state index to insert at.
OCGState	[In] The OCG state to insert.
arrOcgs	[In] The OCG array.

---

**Return**

The number of states inserted.

**Head file reference**

fpd\_docTempl.h: 2135

## FPDActionInsertRendition

### Syntax

```
FS_INT32 FPDActionInsertRendition (
    FPD\_Action action,
    FPD\_Document doc,
    FPD\_Object renditionDict,
    FS_INT32 index
);
```

### Description

Inserts a rendition.

### Parameter

action	[In] A PDF action.
doc	[In] The input PDF document.
renditionDict	[In] The input rendition dictionary.
index	[In] The zero-based rendition index.

### Return

The inserted index in the rendition array.

### Head file reference

fpd\_docTempl.h: 2008

## FPDActionInsertSubAction

### Syntax

```
void FPDActionInsertSubAction (
    FPD\_Action action,
    FS_DWORD index,
    FPD\_Document document,
    const FPD\_Action subAction
);
```

### Description

Inserts a sub-action at specified position.

### Parameter

action	[In] A PDF action.
--------	--------------------

---

index	[In] The zero-based sub-action index to insert at.
-------	--

---

document	[In] The PDF document.
----------	------------------------

---

subAction	[In] The input sub-action.
-----------	----------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2209

**FPDActionIsMixPlay****Syntax**

```
FS_BOOL FPDActionIsMixPlay (
    FPD\_Action action
);
```

**Description**

Checks whether to mix this sound with any other sound already playing.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

**Return**

[TRUE](#) for mixing this sound with any other sound already playing.

**Head file reference**

fpd\_docTempl.h: 2105

**FPDActionIsRepeat****Syntax**

```
FS_BOOL FPDActionIsRepeat (
    FPD\_Action action
);
```

**Description**

Checks whether to repeat the sound indefinitely.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---



**Return**

[TRUE](#) for repeating the sound indefinitely.

**Head file reference**

fpd\_docTempl.h: 2096

**FPDActionIsStatePreserved****Syntax**

```
FS_BOOL FPDActionIsStatePreserved (
    FPD\_Action action
);
```

**Description**

Checks whether the radio-button state relationships between optional content groups apply or not.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

**Return**

[TRUE](#) for the radio-button state relationships between optional content groups applying. Otherwise not.

**Head file reference**

fpd\_docTempl.h: 2170

**FPDActionIsSynchronous****Syntax**

```
FS_BOOL FPDActionIsSynchronous (
    FPD\_Action action
);
```

**Description**

Checks whether to play the sound synchronously or asynchronously.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

**Return**

[TRUE](#) for playing the sound synchronously.

**Head file reference**

fpd\_docTempl.h: 2087

## FPDActionNew

### Syntax

```
FPD_Action FPDActionNew (  
    FPD Object dict  
);
```

### Description

Creates a PDF action from a PDF dictionary.

### Parameter

---

dict	[In] The input PDF dictionary.
------	--------------------------------

---

### Return

A PDF action.

### Head file reference

fpd\_docTempl.h: 1710

## FPDActionNew2

### Syntax

```
FPD_Action FPDActionNew2 (  
    FPD Document doc,  
    FPD ActionType actionType  
);
```

### Description

Creates a PDF action with specified action type.

### Parameter

---

doc	[In] The input PDF document.
-----	------------------------------

---

---

actionType	[In] The input action type.
------------	-----------------------------

---

### Return

A PDF action.

### Head file reference

fpd\_docTempl.h: 1719

## FPDActionNew3

**Syntax**

```
FPD_Action FPDActionNew3 (
    FPD Document doc,
    FS\_LPSTR szType
);
```

**Description**

Creates a PDF action with specified action type name.

**Parameter**

---

doc	[In] The input PDF document.
-----	------------------------------

---

szType	[In] The input action type name.
--------	----------------------------------

---

**Return**

A PDF action.

**Head file reference**

fpd\_docTempl.h: 1729

**FPDActionRemoveAllSubActions****Syntax**

```
void FPDActionRemoveAllSubActions (
    FPD Action action
);
```

**Description**

Removes all sub-actions.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2231

**FPDActionRemoveOCGStates****Syntax**

```
void FPDActionRemoveOCGStates (
    FPD Action action,
    FS\_INT32 index
```

);

**Description**

Removes a OCG state.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

index	[In] The OCG state index.
-------	---------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2160

**FPDActionRemoveRendition****Syntax**

```
void FPDActionRemoveRendition (
    FPD Action action,
    FPD Object renditionDict
);
```

**Description**

Removes a rendition.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

renditionDict	[In] The input rendition dictionary.
---------------	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2020

**FPDActionRemoveSubAction****Syntax**

```
void FPDActionRemoveSubAction (
    FPD Action action,
```



```
FS_DWORD index  
);
```

**Description**

Removes a sub-action.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

index	[In] The zero-based sub-action index.
-------	---------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2221

**FPDActionReplaceOCGStates****Syntax**

```
FS_BOOL FPDActionReplaceOCGStates (  
    FPD_Action action,  
    FPD_Document doc,  
    FS_INT32 iIndex,  
    const FS_PtrArray arrOcgs  
,
```

**Description**

Replaces a OCG state.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

doc	[In] The input PDF document.
-----	------------------------------

---

iIndex	[In] The start OCG state index to replace.
--------	--

---

arrOcgs	[In] The OCG array.
---------	---------------------

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 2148

## FPDActionSetAnnot

### Syntax

```
void FPDActionSetAnnot (
    FPD Action action,
    FPD Document doc,
    FPD Object annotDict
);
```

### Description

Sets the annotation dictionary.

### Parameter

---

action	[In] A PDF action.
--------	--------------------

---

doc	[In] The input PDF document.
-----	------------------------------

---

annotDict	[In] The input annotation dictionary.
-----------	---------------------------------------

---

### Return

void

### Head file reference

fpd\_docTempl.h: 2039

## FPDActionSetDest

### Syntax

```
void FPDActionSetDest (
    FPD Action action,
    const FPD Dest dest
);
```

### Description

Sets the destination.

### Parameter

---

action	[In] A PDF action.
--------	--------------------

---

dest	[In] Ref to a PDF destination.
------	--------------------------------

---



**Return**

void

**Head file reference**

fpd\_docTempl.h: 1778

**FPDActionSetFilePath****Syntax**

```
void FPDActionSetFilePath (
    FPD\_Action action,
    FS\_LPCWSTR wszFilePath,
    FS\_BOOL bIsURL
);
```

**Description**

Sets the file full path.

**Parameter**

action	[In] A PDF action.
wszFilePath	[In] The input file full path.
bIsURL	[In] Whether the file path is a URL or not.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1798

**FPDActionSetFlags****Syntax**

```
void FPDActionSetFlags (
    FPD\_Action action,
    FS\_DWORD dwFlags
);
```

**Description**

Sets the flags for action type SubmitForm, ResetForm.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

dwFlags	[In] The input flags.
---------	-----------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1946

**FPDActionSetHideStatus****Syntax**

```
void FPDActionSetHideStatus (
    FPD\_Action action,
    FS\_BOOL bHide
);
```

**Description**

Sets the hide status of a PDF action.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

bHide	[In] The input hide status.
-------	-----------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1907

**FPDActionSetJavaScript****Syntax**

```
void FPDActionSetJavaScript (
    FPD\_Action action,
    FPD\_Document doc,
    FS\_LPCSTR szJavaScript
);
```

**Description**

Sets the javascript with a byte string.

**Parameter**

---

action	[In] A PDF action.
doc	[In] The input PDF document.
szJavaScript	[In] The javascript in byte string.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1966

**FPDActionSetJavaScriptW****Syntax**

```
void FPDActionSetJavaScriptW (
    FPD\_Action action,
    FPD\_Document doc,
    FS\_LPCWSTR wszJavaScript
);
```

**Description**

Sets the javascript with a wide string.

**Parameter**


---

action	[In] A PDF action.
doc	[In] The input PDF document.
wszJavaScript	[In] The javascript in wide string.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1977

**FPDActionSetMouseMap****Syntax**

```
void FPDActionSetMouseMap (
    FPD\_Action action,
    FS\_BOOL bMap
);
```



**Description**

Sets the mouse-position-tracking flag.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

bMap	[In] The new mouse-position-tracking flag.
------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1878

**FPDActionSetNameAction****Syntax**

```
void FPDActionSetNameAction (
    FPD\_Action action,
    FS\_LPCSTR szName
);
```

**Description**

Sets the name of named action.

**Parameter**

---

action	[In] A PDF action.
--------	--------------------

---

szName	[In] The input name of named action.
--------	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1927

**FPDActionSetNewWindow****Syntax**

```
void FPDActionSetNewWindow (
    FPD\_Action action,
    FS\_BOOL bNewWindow
```

);

**Description**

Sets the new window flag.

**Parameter**

---

action	[In] A PDF action.
bNewWindow	[In] The flag which identifies whether to open the destination document in a new window or not.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1818

**FPDActionSetOperationType****Syntax**

```
void FPDActionSetOperationType (
    FPD Action action,
    FS\_INT32 iType
);
```

**Description**

Sets the operation type.

**Parameter**

---

action	[In] A PDF action.
iType	[In] The input operation type.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2059

**FPDActionSetStatePreserved****Syntax**

```
void FPDActionSetStatePreserved (
```

```
FPD_Action action,  
FS_BOOL bPreserved  
);
```

**Description**

Sets the radio-button state relationships flag.

**Parameter**

---

action	[In] A PDF action.
bPreserved	[In] The new flag.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2179

**FPDActionSetURI****Syntax**

```
void FPDActionSetURI (  
    FPD_Action action,  
    FS_LPSTR szURI  
) ;
```

**Description**

Sets the URI of the PDF action.

**Parameter**

---

action	[In] A PDF action.
szURI	[In] The input URI.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1859

**FPDActionSetWinParam****Syntax**

---

```
void FPDActionSetWinParam (
    FPD\_Action action,
    const FPD\_LWinParam param
);
```

**Description**

Sets the windows launch params.

**Parameter**


---

action	[In] A PDF action.
param	[In] Ref to a windows launch param object.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1838

## FPD\_ActionFields

[Return from Used by](#)

**Description**

FPD\_ActionFields represents the /Fields entry in submit-form actions dictionary. We can use a FPD\_ActionFields object to get field's information easily. See [FPDACTIONFIELDSNEW](#) , [FPDACTIONFIELDSDESTROY](#) , [FPDACTIONGETWIDGETS](#) .

**Returned from**

[FPDACTIONFIELDSNEW](#)

**Used by**

[FPDACTIONGETWIDGETS](#)  
[FPDACTIONFIELDSDESTROY](#)  
[FPDACTIONFIELDSGETACTION](#)  
[FPDACTIONFIELDSGETALLFIELDS](#)  
[FPDACTIONFIELDSGETFIELD](#)  
[FPDACTIONFIELDSGETFIELDSCOUNT](#)  
[FPDACTIONFIELDSINSERTFIELD](#)  
[FPDACTIONFIELDSREMOVEFIELD](#)

**Functions**

[Functions summary](#)

**FPDACTIONFIELDSDESTROY**

Destroys PDF action fields.

**FPDACTIONFIELDSGETACTION**

Gets the PDF action dictionary.

**FPDACTIONFIELDSGETALLFIELDS**

Gets all fields in the action dictionary. outFieldObjects is an array of FPD\_Object.

**FPDACTIONFIELDSGETFIELD**

Gets a field value. returned object may be FPD\_Object or FPD\_String.

**FPDACTIONFIELDSGETFIELDSCOUNT**

Gets the count of action fields.

**FPDACTIONFIELDSINSERTFIELD**

Inserts a field. field: it can be FPD\_Object or FPD\_String object, if need insert field as name(string), construct name as FPD\_String object

**FPDACTIONFIELDSNEW**

Creates PDF action fields from a PDF action.

**FPDACTIONFIELDSREMOVEFIELD**

Removes a field from PDF action fields.

## Functions detail

### FPDACTIONFIELDSDESTROY

#### Syntax

```
void FPDACTIONFIELDSDESTROY (
    FPD_ActionFields actFields
);
```

#### Description

Destroys PDF action fields.

#### Parameter

---

actFields	[In] The input PDF action fields.
-----------	-----------------------------------

---

#### Return

void

#### Head file reference

fpd\_docTempl.h: 1630

### FPDACTIONFIELDSGETACTION

#### Syntax

```
FPD_Action FPDACTIONFIELDSGETACTION (
    FPD_ActionFields actFields
);
```

#### Description

Gets the PDF action dictionary.

**Parameter**

---

actFields	[In] The input PDF action fields.
-----------	-----------------------------------

---

**Return**

The PDF action dictionary.

**Head file reference**

fpd\_docTempl.h: 1693

**FPDActionFields GetAllFields****Syntax**

```
void FPDActionFieldsGetAllFields (
    FPD\_ActionFields actFields,
    FS\_PtrArray* outFieldObjects
);
```

**Description**

Gets all fields in the action dictionary. outFieldObjects is an array of FPD\_Object.

**Parameter**

---

actFields	[In] The input PDF action fields.
-----------	-----------------------------------

---

---

outFieldObjects	[Out] It receives all fields in the action dictionary.
-----------------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1648

**FPDActionFields GetField****Syntax**

```
FPD_Object FPDActionFieldsGetField (
    FPD\_ActionFields actFields,
    FS\_DWORD index
);
```

**Description**

Gets a field value. returned object may be FPD\_Object or FPD\_String.

**Parameter**

---

actFields	[In] The input PDF action fields.
-----------	-----------------------------------

---

index	[In] The zero-based field index.
-------	----------------------------------

---

**Return**

A PDF object.

**Head file reference**

fpd\_docTempl.h: 1639

**FPDActionFieldsGetFieldsCount****Syntax**

```
FS_DWORD FPDActionFieldsGetFieldsCount (
    FPD\_ActionFields actFields
);
```

**Description**

Gets the count of action fields.

**Parameter**

---

actFields	[In] The input PDF action fields.
-----------	-----------------------------------

---

**Return**

The count of action fields.

**Head file reference**

fpd\_docTempl.h: 1639

**FPDActionFieldsInsertField****Syntax**

```
void FPDActionFieldsInsertField (
    FPD\_ActionFields actFields,
    FS\_DWORD iInsertAt,
    const FPD\_Object field
);
```

**Description**

Inserts a field. field: it can be FPD\_Object or FPD\_String object, if need insert field as name(string), construct name as FPD\_String object

**Parameter**

---

actFields	[In] The input PDF action fields.
-----------	-----------------------------------

---

---

iInsertAt	[In] The zero-based field index to insert at.
-----------	---

---

field	[In] The field value.
-------	-----------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1670

**FPDActionFieldsNew****Syntax**

```
FPD_ActionFields FPDActionFieldsNew (
    FPD\_Action action
);
```

**Description**

Creates PDF action fields from a PDF action.

**Parameter**

---

action	[In] The input PDF action.
--------	----------------------------

---

**Return**

PDF action fields.

**Head file reference**

fpd\_docTempl.h: 1621

**FPDActionFieldsRemoveField****Syntax**

```
void FPDActionFieldsRemoveField (
    FPD\_ActionFields actFields,
    FS\_DWORD index
);
```

**Description**

Removes a field from PDF action fields.

**Parameter**

---

actFields	[In] The input PDF action fields.
-----------	-----------------------------------

---

---

index	[In] The zero-based field index to be removed.
-------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1683

## FPD\_AllStates

**Used by****Description**

No document exists.

**Used by**[FPDFFormParseContent](#)[FPDFFormStartParse](#)

## FPD\_Annot

**Return from Used by****Description**

An annotation on a page in a PDF file. See [FPDAnnotNew](#) , [FPDAnnotDestroy](#) [FRAAnnotGetPDAAnnot](#) , [FPDAnnotListGetAt](#) .

**Returned from**[FRAAnnotGetPDFAnnot](#)[FPDAnnotListGetAt](#)[FPDAnnotListNew](#)[FPDAnnotGetIRTNote](#)[FPDAnnotNew](#)**Used by**[FPDAnnotListCount](#)[FPDAnnotListDestroy](#)[FPDAnnotListDisplayAnnots](#)[FPDAnnotListDisplayAnnotsEx](#)[FPDAnnotListGetAnnotMatrix](#)[FPDAnnotListGetAnnotRect](#)[FPDAnnotListGetAt](#)[FPDAnnotListGetDocument](#)[FPDAnnotListGetIndex](#)[FPDAnnotListInsert](#)

[FPDAnotListMoveTo](#)  
[FPDAnotListMoveToFirst](#)  
[FPDAnotListMoveToLast](#)  
[FPDAnotListRemove](#)  
[FPDAnotListRemoveAll](#)  
[FPDAnotListRemoveTemp](#)  
[FPDAnotListReplace](#)  
[FPDAnotListSetFixedIconParams](#)  
[FPDFFormControlDrawControl](#)  
[FPDAnotClearCachedAP](#)  
[FPDAnotCountIRTNotes](#)  
[FPDAnotDestroy](#)  
[FPDAnotDrawAppearance](#)  
[FPDAnotDrawBorder](#)  
[FPDAnotDrawInContext](#)  
[FPDAnotGetAnnotDict](#)  
[FPDAnotGetAPForm](#)  
[FPDAnotGetFlags](#)  
[FPDAnotGetIRTNote](#)  
[FPDAnotGetRect](#)  
[FPDAnotGetSubType](#)

## Definitions

### Definitions summary

#### [FPD ANNOTFLAG\\_HIDDEN](#)

If set, do not display or print the annotation or allow it to interact with the user, regardless of its annotation type or whether an annotation handler is available.

#### [FPD ANNOTFLAG\\_INVISIBLE](#)

If set, do not display the annotation if it does not belong to one of the standard annotation types and no annotation handler is available. If clear, display such an unknown annotation using an appearance stream specified by its appearance dictionary, if any.

#### [FPD ANNOTFLAG\\_LOCKED](#)

If set, do not allow the annotation to be deleted or its properties (including position and size) to be modified by the user.

#### [FPD ANNOTFLAG\\_NORotate](#)

If set, do not rotate the annotation's appearance to match the rotation of the page.

#### [FPD ANNOTFLAG\\_NOVIEW](#)

If set, do not display the annotation on the screen or allow it to interact with the user.

#### [FPD ANNOTFLAG\\_NOZOOM](#)

If set, do not scale the annotation's appearance to match the magnification of the page.

#### [FPD ANNOTFLAG\\_PRINT](#)

If set, print the annotation when the page is printed. If clear, never print the annotation, regardless of whether it is displayed on the screen.

#### [FPD ANNOTFLAG\\_READONLY](#)

If set, do not allow the annotation to interact with the user.

#### [FPD ANNOTFLAG\\_TOGGLENOVIEW](#)

If set, invert the interpretation of the NoView flag for certain events.

### Definitions detail

#### FPD\_ANNOTFLAG\_HIDDEN

**Syntax**

```
#define FPD_ANNOTFLAG_HIDDEN 2
```

**Description**

If set, do not display or print the annotation or allow it to interact with the user, regardless of its annotation type or whether an annotation handler is available.

**Group**

[FPDAnnotationFlagBits](#)

**Head file reference**

fpd\_docExpT.h: 694

## FPD\_ANNOTFLAG\_INVISIBLE

**Syntax**

```
#define FPD_ANNOTFLAG_INVISIBLE 1
```

**Description**

If set, do not display the annotation if it does not belong to one of the standard annotation types and no annotation handler is available. If clear, display such an unknown annotation using an appearance stream specified by its appearance dictionary, if any.

**Group**

[FPDAnnotationFlagBits](#)

**Head file reference**

fpd\_docExpT.h: 688

## FPD\_ANNOTFLAG\_LOCKED

**Syntax**

```
#define FPD_ANNOTFLAG_LOCKED 0x80
```

**Description**

If set, do not allow the annotation to be deleted or its properties (including position and size) to be modified by the user.

**Group**

[FPDAnnotationFlagBits](#)

**Head file reference**

fpd\_docExpT.h: 719

## FPD\_ANNOTFLAG\_NORotate

**Syntax**

```
#define FPD_ANNOTFLAG_NORotate 0x10
```

**Description**

If set, do not rotate the annotation's appearance to match the rotation of the page.

**Group**

[FPDAnnotationFlagBits](#)

**Head file reference**

fpd\_docExpT.h: 707

## FPD\_ANNOTFLAG\_NOVIEW

**Syntax**

```
#define FPD_ANNOTFLAG_NOVIEW 0x20
```

**Description**

If set, do not display the annotation on the screen or allow it to interact with the user.

**Group**

[FPDAnnotationFlagBits](#)

**Head file reference**

fpd\_docExpT.h: 711

## FPD\_ANNOTFLAG\_NOZOOM

**Syntax**

```
#define FPD_ANNOTFLAG_NOZOOM 8
```

**Description**

If set, do not scale the annotation's appearance to match the magnification of the page.

**Group**

[FPDAnnotationFlagBits](#)

**Head file reference**

fpd\_docExpT.h: 703

## FPD\_ANNOTFLAG\_PRINT

**Syntax**

```
#define FPD_ANNOTFLAG_PRINT 4
```

**Description**

If set, print the annotation when the page is printed. If clear, never print the annotation, regardless of whether it is displayed on the screen.

**Group**[FPDAnnotationFlagBits](#)**Head file reference**

fpd\_docExpT.h: 699

**FPD\_ANNOTFLAG\_READONLY****Syntax**

#define FPD\_ANNOTFLAG\_READONLY 0x40

**Description**

If set, do not allow the annotation to interact with the user.

**Group**[FPDAnnotationFlagBits](#)**Head file reference**

fpd\_docExpT.h: 715

**FPD\_ANNOTFLAG\_TOGGLENOVIEW****Syntax**

#define FPD\_ANNOTFLAG\_TOGGLENOVIEW 0x100

**Description**

If set, invert the interpretation of the NoView flag for certain events.

**Group**[FPDAnnotationFlagBits](#)**Head file reference**

fpd\_docExpT.h: 723

## Enumerations

### Enumerations summary

**[FPD\\_AnnotAppearanceMode](#)**Appearance mode enumeration of annotation. See [FPDAnnotDrawAppearance](#) , [FPDAnnotGetAPForm](#) , [FPDFFormControlDrawControl](#) .

### Enumerations detail

**FPD\_AnnotAppearanceMode****Syntax**

```
enum FPD_AnnotAppearanceMode{
    NormalAppearanceMode,
    RolloverAppearanceMode,
    DownAppearanceMode
};
```

### Description

Apearance mode enumeration of annotation. See [FPDAnnotDrawAppearance](#) , [FPDAnnotGetAPForm](#) , [FPDFFormControlDrawControl](#) .

### Head file reference

fpd\_docExpT.h: 733

#### **NormalAppearanceMode**

Used when the annotation is not interacting with the user. This appearance is also used for printing the annotation.

#### **RolloverAppearanceMode**

Used when the user moves the cursor into the annotation's active area without pressing the mouse button.

#### **DownAppearanceMode**

Used when the mouse button is pressed or held down within the annotation's active area.

## Functions

### Functions summary

#### [\*\*FPDAnnotClearCachedAP\*\*](#)

Clears all cached appearance, when the application changed any appearance settings.

#### [\*\*FPDAnnotCountIRTNotes\*\*](#)

Gets the count of annotations in "in reply to" annotation list.

#### [\*\*FPDAnnotDestroy\*\*](#)

Destroys an annotation.

#### [\*\*FPDAnnotDrawAppearance\*\*](#)

Draws annotation's appearance, using default appearance rules.

#### [\*\*FPDAnnotDrawBorder\*\*](#)

Draws border of the annotation, using border settings within the dictionary.

#### [\*\*FPDAnnotDrawInContext\*\*](#)

Draws annotation's appearance in rendering context.

#### [\*\*FPDAnnotGetAnnotDict\*\*](#)

Gets the annotation dictionary. We store all information within the original annotation dictionary.

#### [\*\*FPDAnnotGetAPForm\*\*](#)

Gets an appearance stream.

#### [\*\*FPDAnnotGetFlags\*\*](#)

Gets the annotation flags.

#### [\*\*FPDAnnotGetIRTNote\*\*](#)

Gets an "in reply to" annotation.

#### [\*\*FPDAnnotGetRect\*\*](#)

Gets annotation bounding box in user space.

**[FPDAnnotGetSubType](#)**

Gets the annotation type name.

**[FPDAnnotNew](#)**

Creates an annotation from a PDF dictionary.

## Functions detail

**FPDAnnotClearCachedAP****Syntax**

```
void FPDAnnotClearCachedAP (
    FPD\_Annot annot
);
```

**Description**

Clears all cached appearance, when the application changed any appearance settings.

**Parameter**

---

annot	[In] The input annotation.
-------	----------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3389

**FPDAnnotCountIRTNotes****Syntax**

```
FS_INT32 FPDAnnotCountIRTNotes (
    FPD\_Annot annot
);
```

**Description**

Gets the count of annotations in "in reply to" annotation list.

**Parameter**

---

annot	[In] The input annotation.
-------	----------------------------

---

**Return**

The count of annotations in "in reply to" annotation list.

**Head file reference**

fpd\_docTempl.h: 3410

## FPDAnnotDestroy

### Syntax

```
void FPDAnnotDestroy (
    FPD\_Annot annot
);
```

### Description

Destroys an annotation.

### Parameter

---

annot	[In] The input annotation.
-------	----------------------------

---

### Return

void

### Head file reference

fpd\_docTempl.h: 3316

## FPDAnnotDrawAppearance

### Syntax

```
FS_BOOL FPDAnnotDrawAppearance (
    annot,
    const FPD\_Page page,
    FPD\_RenderDevice dc,
    FS\_AffineMatrix matrix,
    FPD\_AnnotAppearanceMode eMode,
    const FPD\_RenderOptions opts
);
```

### Description

Draws annotation's appearance, using default appearance rules.

### Parameter

---

annot	[In] The input annotation.
-------	----------------------------

---

---

page	[In] The page it belongs to.
------	------------------------------

---

---

dc	[In] The device to draw on.
----	-----------------------------

---

---

matrix	[In] The transformation matrix from user space to device space.
--------	---

---

---

eMode	[In] The input appearance mode.
-------	---------------------------------

---

opts	[In] The render options.
------	--------------------------

---

**Return**

[TRUE](#) if the appearance successfully found and drawn.

**Head file reference**

fpd\_docTempl.h: 3362

**FPDAnnotDrawBorder****Syntax**

```
void FPDAnnotDrawBorder (
    FPD\_Annot annot,
    FPD\_RenderDevice dc,
    FS\_AffineMatrix matrix,
    const FPD\_RenderOptions opts
);
```

**Description**

Draws border of the annotation, using border settings within the dictionary.

**Parameter**

---

annot	[In] The input annotation.
-------	----------------------------

---

dc	[In] The device to draw on.
----	-----------------------------

---

matrix	[In] The transformation matrix from user space to device space.
--------	---

---

opts	[In] The render options.
------	--------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3398

**FPDAnnotDrawInContext****Syntax**

```
FS_BOOL FPDAnnotDrawInContext (
    FPD\_Annot annot,
    const FPD\_Page page,
```

---

```
const FPD_RenderContext context,
FS_AffineTransform matrix,
FPD_AnnotAppearanceMode eMode
);
```

**Description**

Draws annotation's appearance in rendering context.

**Parameter**

annot	[In] The input annotation.
page	[In] The page it belongs to.
context	[In] The context to draw in.
matrix	[In] The transformation matrix from user space to device space.
eMode	[In] The input appearance mode.

**Return**

[TRUE](#) if the appearance successfully found and drawn.

**Head file reference**

fpd\_docTempl.h: 3376

**FPDAnnotGetAnnotDict****Syntax**

```
FPD_Object FPDAnnotGetAnnotDict (
    FPD_Annot annot
);
```

**Description**

Gets the annotation dictionary. We store all information within the original annotation dictionary.

**Parameter**

annot	[In] The input annotation.
-------	----------------------------

**Return**

The annotation dictionary.

**Head file reference**

fpd\_docTempl.h: 3325

## FPDAnnotGetAPForm

### Syntax

```
FPD_Form FPDAnnotGetAPForm (
    FPD\_Annot annot,
    const FPD\_Page page,
    FPD\_AnnotAppearanceMode eMode
);
```

### Description

Gets an appearance stream.

### Parameter

---

annot	[In] The input annotation.
-------	----------------------------

---

page	[In] The page it belongs to.
------	------------------------------

---

eMode	[In] The input appearance mode.
-------	---------------------------------

---

### Return

A PDF form.

### Head file reference

fpd\_docTempl.h: 3429

## FPDAnnotGetFlags

### Syntax

```
FS_DWORD FPDAnnotGetFlags (
    FPD\_Annot annot
);
```

### Description

Gets the annotation flags.

### Parameter

---

annot	[In] The input annotation.
-------	----------------------------

---

### Return

The annotation flags.

### Head file reference



fpd\_docTempl.h: 3344

## FPDAnnotGetIRTNote

### Syntax

```
FPD_Annot FPDAnnotGetIRTNote (
    FPD\_Annot annot,
    FS\_INT32 index
);
```

### Description

Gets an "in reply to" annotation.

### Parameter

---

annot	[In] The input annotation.
index	[In] The zero-based index in the "in reply to" annotation list.

---

### Return

An "in reply to" annotation.

### Head file reference

fpd\_docTempl.h: 3419

## FPDAnnotGetRect

### Syntax

```
FS_FloatRect FPDAnnotGetRect (
    FPD\_Annot annot
);
```

### Description

Gets annotation bounding box in user space.

### Parameter

---

annot	[In] The input annotation.
-------	----------------------------

---

### Return

The bounding box of the annotation.

### Head file reference

fpd\_docTempl.h: 3353

## FPDAnnotGetSubType

### Syntax

```
void FPDAnnotGetSubType (
    FPD\_Annot annot,
    FS\_ByteString* outSubType
);
```

### Description

Gets the annotation type name.

### Parameter

---

annot	[In] The input annotation.
-------	----------------------------

---

outSubType	[Out] It receives the annotation type name.
------------	---

---

### Return

void

### Head file reference

fpd\_docTempl.h: 3334

## FPDAnnotNew

### Syntax

```
FPD_Annot FPDAnnotNew (
    FPD\_Object dict
);
```

### Description

Creates an annotation from a PDF dictionary.

### Parameter

---

dict	[In] The input PDF dictionary.
------	--------------------------------

---

### Return

An annotation.

### Head file reference

fpd\_docTempl.h: 3307

## FPD\_AnnotList

## **Return from Used by**

### Description

An annotation accessor. See [FPDAnnotListNew](#) , [FPDAnnotListDestroy](#) .

### Returned from

#### [\*\*FPDAnnotListNew\*\*](#)

### Used by

[\*\*FPDAnnotListCount\*\*](#)  
[\*\*FPDAnnotListDestroy\*\*](#)  
[\*\*FPDAnnotListDisplayAnnots\*\*](#)  
[\*\*FPDAnnotListDisplayAnnotsEx\*\*](#)  
[\*\*FPDAnnotListGetAnnotMatrix\*\*](#)  
[\*\*FPDAnnotListGetAnnotRect\*\*](#)  
[\*\*FPDAnnotListGetAt\*\*](#)  
[\*\*FPDAnnotListGetDocument\*\*](#)  
[\*\*FPDAnnotListGetIndex\*\*](#)  
[\*\*FPDAnnotListInsert\*\*](#)  
[\*\*FPDAnnotListMoveTo\*\*](#)  
[\*\*FPDAnnotListMoveToFirst\*\*](#)  
[\*\*FPDAnnotListMoveToLast\*\*](#)  
[\*\*FPDAnnotListRemove\*\*](#)  
[\*\*FPDAnnotListRemoveAll\*\*](#)  
[\*\*FPDAnnotListRemoveTemp\*\*](#)  
[\*\*FPDAnnotListReplace\*\*](#)  
[\*\*FPDAnnotListSetFixedIconParams\*\*](#)

### Functions

#### Functions summary

##### [\*\*FPDAnnotListCount\*\*](#)

Gets the count of annotations in the annotation list.

##### [\*\*FPDAnnotListDestroy\*\*](#)

Destroys an annotation list.

##### [\*\*FPDAnnotListDisplayAnnots\*\*](#)

Display all annotations with specified object to DC transformation.

##### [\*\*FPDAnnotListDisplayAnnotsEx\*\*](#)

Adds annotations to render job list, used in printing for transparency support.

##### [\*\*FPDAnnotListGetAnnotMatrix\*\*](#)

Gets annotation transformation matrix from page space to device space.

##### [\*\*FPDAnnotListGetAnnotRect\*\*](#)

Gets annotation rectangle in device coordinates.

##### [\*\*FPDAnnotListGetAt\*\*](#)

Gets an annotation at specified position.

##### [\*\*FPDAnnotListGetDocument\*\*](#)

Gets the PDF document.

##### [\*\*FPDAnnotListGetIndex\*\*](#)

Gets the annotation's index.

**FPDAnnotListInsert**

Inserts BEFORE the index, if index equals to current count, then append.

**FPDAnnotListMoveTo**

Moves an annotation to specified position.

**FPDAnnotListMoveToFirst**

Moves an annotation to the first of the annotation list.

**FPDAnnotListMoveToLast**

Moves an annotation to the last of the annotation list.

**FPDAnnotListNew**

Creates an annotation list from a PDF page.

**FPDAnnotListRemove**

Removes an annotation.

**FPDAnnotListRemoveAll**

Removes all the annotations.

**FPDAnnotListRemoveTemp**

Removes from list only, not from page.

**FPDAnnotListReplace**

Replaces the specified position with a new annotation.

**FPDAnnotListSetFixedIconParams**

Enable to show icon annotations as fixed size (NoZoom or NoRotate flag).

## Functions detail

### FPDAnnotListCount

**Syntax**

```
FS_INT32 FPDAnnotListCount (
    FPD_AnnotList annotList
);
```

**Description**

Gets the count of annotations in the annotation list.

**Parameter**

---

annotList	[In] The input annotation list.
-----------	---------------------------------

---

**Return**

The count of annotations in the annotation list.

**Head file reference**

fpd\_docTempl.h: 3505

### FPDAnnotListDestroy

**Syntax**

```
void FPDAnnotListDestroy (
    FPD_AnnotList annotList
```

);

**Description**

Destroys an annotation list.

**Parameter**

---

annotList	[In] The input annotation list.
-----------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3457

**FPDAnnotListDisplayAnnots****Syntax**

```
void FPDAnnotListDisplayAnnots (
    FPD\_AnnotList annotList,
    const FPD\_Page page,
    FPD\_RenderDevice dc,
    FS\_AffineMatrix matrix,
    FS\_BOOL bShowWidget,
    FPD\_RenderOptions opts
);
```

**Description**

Display all annotations with specified object to DC transformation.

**Parameter**

---

annotList	[In] The input annotation list.
-----------	---------------------------------

---

---

page	[In] The page it belongs to.
------	------------------------------

---

---

dc	[In] The device to draw on.
----	-----------------------------

---

---

matrix	[In] The transformation matrix from objects space to device space.
--------	--

---

---

bShowWidget	[In] Whether to show widget or not.
-------------	-------------------------------------

---

---

opts	[In] The render options.
------	--------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3466

**FPDAnnotListDisplayAnnotsEx****Syntax**

```
void FPDAnnotListDisplayAnnotsEx (  
    FPD\_AnnotList annotList,  
    const FPD\_Page pPage,  
    FPD\_RenderContext context,  
    FS\_BOOL bPrinting,  
    FS\_AffineMatrix matrix,  
    FS\_BOOL bShowWidget,  
    FPD\_RenderOptions opts  
) ;
```

**Description**

Adds annotations to render job list, used in printing for transparency support.

**Parameter**

---

annotList	[In] The input annotation list.
-----------	---------------------------------

---

pPage	[In] The page it belongs to.
-------	------------------------------

---

context	[In] The context to draw in.
---------	------------------------------

---

bPrinting	[In] Whether to print or not.
-----------	-------------------------------

---

matrix	[In] The transformation matrix from objects space to device space.
--------	--

---

bShowWidget	[In] Whether to show widget or not.
-------------	-------------------------------------

---

opts	[In] The render options.
------	--------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3480

## FPDAnnotListGetAnnotMatrix

### Syntax

```
void FPDAnnotListGetAnnotMatrix (
    FPD\_AnnotList annotList,
    FPD\_Object pAnnotDict,
    FS\_AffineMatrix pUser2Device,
    FS\_AffineMatrix* outMatrix
);
```

### Description

Gets annotation transformation matrix from page space to device space.

### Parameter

---

annotList	[In] The input annotation list.
-----------	---------------------------------

---

pAnnotDict	[In] Annotation's dictionary object.
------------	--------------------------------------

---

pUser2Device	[In] Current displaying transformation matrix.
--------------	--

---

outMatrix	[Out] Annotation matrix to display.
-----------	-------------------------------------

---

### Return

void.

### Head file reference

[fpd\\_docTempl.h: 3633](#)

### Since

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

## FPDAnnotListGetAnnotRect

### Syntax

```
void FPDAnnotListGetAnnotRect (
    FPD\_AnnotList annotList,
    FPD\_Object pAnnotDict,
    FS\_AffineMatrix pUser2Device,
    FS\_FloatRect* outAnnotRC
);
```

### Description

Gets annotation rectangle in device coordinates.

**Parameter**


---

annotList	[In] The input annotation list.
pAnnotDict	[In] Annotation's dictionary object.
pUser2Device	[In] Transformation matrix from user space to device space.
outAnnotRC	[Out] Annotation rectangle in device coordinates.

---

**Return**

void.

**Head file reference**

fpd\_docTempl.h: 3646

**Since**[SDK LATEEST VERSION > 7.1.0.0](#)**FPDAnnotListGetAt****Syntax**

```
FPD_Annot FPDAnnotListGetAt (
    FPD\_AnnotList annotList,
    FS\_INT32 index
);
```

**Description**

Gets an annotation at specified position.

**Parameter**


---

annotList	[In] The input annotation list.
index	[In] The zero-based index in the annotation list.

---

**Return**

An annotation.

**Head file reference**

fpd\_docTempl.h: 3495

**FPDAnnotListGetDocument**

**Syntax**

```
FPD_Document FPDAnnotListGetDocument (
    FPD\_AnnotList annotList
);
```

**Description**

Gets the PDF document.

**Parameter**

---

annotList	[In] The input annotation list.
-----------	---------------------------------

---

**Return**

The PDF document.

**Head file reference**

fpd\_docTempl.h: 3607

**FPDAnnotListGetIndex****Syntax**

```
FS_INT32 FPDAnnotListGetIndex (
    FPD\_AnnotList annotList,
    FPD\_Annot annot
);
```

**Description**

Gets the annotation's index.

**Parameter**

---

annotList	[In] The input annotation list.
-----------	---------------------------------

---

---

annot	[In] The input annotation.
-------	----------------------------

---

**Return**

The zero-based index in the annotation list.

**Head file reference**

fpd\_docTempl.h: 3514

**FPDAnnotListInsert****Syntax**

```
void FPDAnnotListInsert (
    FPD\_AnnotList annotList,
    FS\_INT32 index,
```

---

```
FPD_Annot annot
);
```

**Description**

Inserts BEFORE the index, if index equals to current count, then append.

**Parameter**

annotList	[In] The input annotation list.
index	[In] The zero-based index to insert before.
annot	[In] The annotation to insert.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3545

**FPDAnnotListMoveTo****Syntax**

```
void FPDAnnotListMoveTo (
    FPD_AnnotList annotList,
    FPD_Annot annot,
    FS_INT32 iNewIndex
);
```

**Description**

Moves an annotation to specified position.

**Parameter**

annotList	[In] The input annotation list.
annot	[In] The annotation to be moved.
iNewIndex	[In] The new position to move to.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3576

## FPDAnnotListMoveToFirst

### Syntax

```
void FPDAnnotListMoveToFirst (
    FPD\_AnnotList annotList,
    FS\_INT32 index
);
```

### Description

Moves an annotation to the first of the annotation list.

### Parameter

---

annotList	[In] The input annotation list.
index	[In] The zero-based index of the annotation to be moved.

---

### Return

void

### Head file reference

fpd\_docTempl.h: 3576

## FPDAnnotListMoveToLast

### Syntax

```
void FPDAnnotListMoveToLast (
    FPD\_AnnotList annotList,
    FS\_INT32 index
);
```

### Description

Moves an annotation to the last of the annotation list.

### Parameter

---

annotList	[In] The input annotation list.
index	[In] The zero-based index of the annotation to be moved.

---

### Return

void

**Head file reference**

fpd\_docTempl.h: 3586

**FPDAnnotListNew****Syntax**

```
FPD_AnnotList FPDAnnotListNew (
    FPD\_Page page
);
```

**Description**

Creates an annotation list from a PDF page.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

An annotation list.

**Head file reference**

fpd\_docTempl.h: 3448

**FPDAnnotListRemove****Syntax**

```
void FPDAnnotListRemove (
    FPD\_AnnotList annotList,
    FS\_INT32 index
);
```

**Description**

Removes an annotation.

**Parameter**

---

annotList	[In] The input annotation list.
-----------	---------------------------------

---

---

index	[In] The zero-based index of the annotation to be removed.
-------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3524

## FPDAnnotListRemoveAll

### Syntax

```
void FPDAnnotListRemoveAll (
    FPD\_AnnotList annotList
);
```

### Description

Removes all the annotations.

### Parameter

---

annotList	[In] The input annotation list.
-----------	---------------------------------

---

### Return

void

### Head file reference

fpd\_docTempl.h: 3524

**Note:** Operations to the annotation list will be reflected in the PDF document structure. For example, the "Annots" array of page dictionary will get changed.

## FPDAnnotListRemoveTemp

### Syntax

```
void FPDAnnotListRemoveTemp (
    FPD\_AnnotList annotList,
    FS\_INT32 index
);
```

### Description

Removes from list only, not from page.

### Parameter

---

annotList	[In] The input annotation list.
-----------	---------------------------------

---

index	[In] The zero-based index of the annotation to be removed.
-------	--

---

### Return

void

### Head file reference

fpd\_docTempl.h: 3566

**FPDAnnotListReplace****Syntax**

```
void FPDAnnotListReplace (
    FPD\_AnnotList annotList,
    FS\_INT32 index,
    FPD\_Annot annot
);
```

**Description**

Replaces the specified position with a new annotation.

**Parameter**

annotList	[In] The input annotation list.
index	[In] The zero-based index in the annotation list.
annot	[In] The new annotation.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3534

**FPDAnnotListSetFixedIconParams****Syntax**

```
void FPDAnnotListSetFixedIconParams (
    FPD\_AnnotList annotList,
    FS\_INT32 nFixedIconType,
    FS\_FLOAT sx,
    FS\_FLOAT sy,
    FS\_INT32 nRotate
);
```

**Description**

Enable to show icon annotations as fixed size (NoZoom or NoRotate flag).

**Parameter**

annotList	[In] The input annotation list.
nFixedIconType	[In] Indicates how to display fixed annotations. 0 is to show as normal annotations, ignore NoZoom or NoRotate flag, and 1 is to show as fixed scaling size, sx and sy specifies scaling

---

fractors in x and y axes, and 2 is to show as fixed device size which is specified by *sx* and *sy* in device units.

---

**sx** [In] Fixed scaling size or fixed device size according to the value of *nFixedIconType* in x axis.

---

**sy** [In] Fixed scaling size or fixed device size according to the value of *nFixedIconType* in y axis.

---

**nRotate** [In] Page rotation, valid values are 0, 1, 2, 3, same as [FPDPageGetDisplayMatrix](#).

---

**Return**

The PDF document.

**Head file reference**

fpd\_docTempl.h: 3616

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1.0.0](#)

## FPD\_Array

### Description

A FPD\_Array is one-dimensional collections of objects accessed by a numeric index. Array indexes are zero-based. An array's elements may be any combination of the FPD\_Object types.

### Functions

#### Functions summary

**[FPDArrayAdd](#)**

Adds a new element to array.

**[FPDArrayAddInteger](#)**

Adds a number object with an integer value.

**[FPDArrayAddName](#)**

Adds a name object.

**[FPDArrayAddNumber](#)**

Adds a number object with a FIX24.8 value.

**[FPDArrayAddReference2ToDoc](#)**

Adds a reference object with object pointer.

**[FPDArrayAddReference2ToFDFDoc](#)**

Adds a reference object with object pointer.

**[FPDArrayAddReferenceToDoc](#)**

Adds a reference object with object number.

**[FPDArrayAddReferenceToFDFDoc](#)**

Adds a reference object with object number.

**[FPDArrayAddString](#)**

Adds a string object.

**[FPDArrayGetArray](#)**

Gets an array object with specified position.

**[FPDArrayGetCount](#)**

Gets the count of objects in the array.

**[FPDArrayGetDict](#)**

Gets a dictionary object with specified position.

**[FPDArrayGetElement](#)**

Gets reference to element. Returns direct reference to the element.

**[FPDArrayGetElementValue](#)**

Gets direct or referred indirect object.

**[FPDArrayGetFloat](#)**

Gets a floating-point with specified position.

**[FPDArrayGetInteger](#)**

Gets an integer with specified position.

**[FPDArrayGetMatrix](#)**

Gets a matrix from the array.

**[FPDArrayGetNumber](#)**

Gets a number with specified position.

**[FPDArrayGetRect](#)**

Gets a rectangle from the array.

**[FPDArrayGetStream](#)**

Gets a stream object with specified position.

**[FPDArrayGetString](#)**

Gets a string with specified position.

**[FPDArrayInsertAt](#)**

Inserts an element at specified position.

**[FPDArrayIsIdentical](#)**

Compares with another object.

**[FPDArrayNew](#)**

Creates an empty array object.

**[FPDArrayRemoveAt](#)**

Removes an element.

**[FPDArraySetAt](#)**

Changes the element at specified position.

## Functions detail

### FPDArrayAdd

#### Syntax

```
void FPDArrayAdd (
    FPD Object objArray,
    FPD Object other_obj,
    void* objs
);
```

#### Description

---

Adds a new element to array.

#### Parameter

objArray	[In] The input PDF array object.
other_obj	[In] The input object.
objs	[In] The indirect object collection, it can be a FDF_Document object or a FPD_Document object, required if pObj is an indirect object. In this case, a reference object will be created and inserted into the array.

---

#### Return

void

#### Head file reference

fpd\_objsTempl.h: 689

**Note:** All elements will be managed with the array object, so the object pointer must NOT be freed by caller

#### Since

[SDK LATEEST VERSION > 1.0](#)

## FPDArrayAddInteger

#### Syntax

```
void FPDArrayAddInteger (
    FPD Object objArray,
    FS\_INT32 value
);
```

#### Description

Adds a number object with an integer value.

#### Parameter

objArray	[In] The input PDF array object.
value	[In] The input integer value.

---

#### Return

void

#### Head file reference

fpd\_objsTempl.h: 712

### FPDArrayAddName

#### Syntax

```
void FPDArrayAddName (
    FPD Object objArray,
    FS LPCSTR szName
);
```

#### Description

Adds a name object.

#### Parameter

---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

szName	[In] The input name data.
--------	---------------------------

---

#### Return

void

#### Head file reference

fpd\_objsTempl.h: 733

### FPDArrayAddNumber

#### Syntax

```
void FPDArrayAddNumber (
    FPD Object objArray,
    FS FLOAT num
);
```

#### Description

Adds a number object with a FIX24.8 value.

#### Parameter

---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

num	[In] The input FIX24.8 value.
-----	-------------------------------

---

#### Return

void

**Head file reference**

fpd\_objsTempl.h: 702

**FPDArrayAddReference2ToDoc****Syntax**

```
void FPDArrayAddReference2ToDoc (
    FPD Object objArray,
    FPD Document doc,
    FPD Object obj
);
```

**Description**

Adds a reference object with object pointer.

**Parameter**

---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

doc	[In] The input indirect object collection.
-----	--

---

obj	[In] The input object.
-----	------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 765

**FPDArrayAddReference2ToFDFDoc****Syntax**

```
void FPDArrayAddReference2ToFDFDoc (
    FPD Object objArray,
    FDF Document doc,
    FPD Object obj
);
```

**Description**

Adds a reference object with object pointer.

**Parameter**

---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

doc	[In] The input indirect object collection.
-----	--

---

---

obj	[In] The input object.
-----	------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 776

**FPDArrayAddReferenceToDoc****Syntax**

```
void FPDArrayAddReferenceToDoc (
    FPD Object objArray,
    FPD Document doc,
    FS DWORD objNum
);
```

**Description**

Adds a reference object with object number.

**Parameter**

---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

---

doc	[In] The input indirect object collection.
-----	--

---

---

objNum	[In] The referred object number.
--------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 743

**FPDArrayAddReferenceToFDFDoc****Syntax**

```
void FPDArrayAddReferenceToFDFDoc (
    FPD Object objArray,
    FDF Document doc,
    FS DWORD objNum
);
```

**Description**

Adds a reference object with object number.

**Parameter**


---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

doc	[In] The input indirect object collection.
-----	--

---

objNum	[In] The referred object number.
--------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 754

**FPDArrayAddString****Syntax**

```
void FPDArrayAddString (
    FPD Object objArray,
    FS LPCSTR str,
    FS INT32 nLen
);
```

**Description**

Adds a string object.

**Parameter**


---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

str	[In] The input string data.
-----	-----------------------------

---

nLen	[In] The length of the input string data.
------	---

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 722

**FPDArrayGetArray****Syntax**

```
FPD_Object FPDArrayGetArray (
    FPD Object objArray,
```

```
FS_DWORD index  
);
```

**Description**

Gets an array object with specified position.

**Parameter**

objArray	[In] The input PDF array object.
index	[In] Specifies the zero-based index in the array.

**Return**

An array object.

**Head file reference**

fpd\_objsTempl.h: 633

**FPDArrayGetCount****Syntax**

```
FS_DWORD FPDArrayGetCount (  
    FPD_Object objArray  
);
```

**Description**

Gets the count of objects in the array.

**Parameter**

objArray	[In] The input PDF array object.
----------	----------------------------------

**Return**

The count of objects in the array.

**Head file reference**

fpd\_objsTempl.h: 535

**FPDArrayGetDict****Syntax**

```
FPD_Object FPDArrayGetDict (  
    FPD_Object objArray,  
    FS_DWORD index  
);
```

**Description**

Gets a dictionary object with specified position.

**Parameter**

objArray	[In] The input PDF array object.
index	[In] Specifies the zero-based index in the array.

**Return**

A dictionary object.

**Head file reference**

fpd\_objsTempl.h: 613

**FPDArrayGetElement****Syntax**

```
FPD_Object FPDArrayGetElement (
    FPD Object objArray,
    FS\_DWORD index
);
```

**Description**

Gets reference to element. Returns direct reference to the element.

**Parameter**

objArray	[In] The input PDF array object.
index	[In] Specifies the zero-based index in the array.

**Return**

Pointer to specified element.

**Head file reference**

fpd\_objsTempl.h: 544

**Note:** Don't release the returned object.

**FPDArrayGetValue****Syntax**

```
FPD_Object FPDArrayGetValue (
```

```
FPD_Object objArray,  
FS_DWORD index  
);
```

**Description**

Gets direct or referred indirect object.

**Parameter**

---

objArray	[In] The input PDF array object.
index	[In] Specifies the zero-based index in the array.

---

**Return**

A direct or referred indirect object.

**Head file reference**

fpd\_objsTempl.h: 554

**Note:** Don't release the returned object.

**FPDArrayGetFloat****Syntax**

```
FS_FLOAT FPDArrayGetFloat (  
    FPD_Object objArray,  
    FS_DWORD index  
) ;
```

**Description**

Gets a floating-point with specified position.

**Parameter**

---

objArray	[In] The input PDF array object.
index	[In] Specifies the zero-based index in the array.

---

**Return**

A floating-point value.

**Head file reference**

fpd\_objsTempl.h: 643

## FPDArrayGetInteger

### Syntax

```
FS_INT32 FPDArrayGetInteger (
    FPD Object objArray,
    FS\_DWORD index
);
```

### Description

Gets an integer with specified position.

### Parameter

---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

index	[In] Specifies the zero-based index in the array.
-------	---

---

### Return

An integer.

### Head file reference

fpd\_objsTempl.h: 593

## FPDArrayGetMatrix

### Syntax

```
FS_AffineMatrix FPDArrayGetMatrix (
    FPD Object objArray
);
```

### Description

Gets a matrix from the array.

### Parameter

---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

### Return

A matrix from the array.

### Head file reference

fpd\_objsTempl.h: 564

## FPDArrayGetNumber

### Syntax

```
FS_FLOAT FPDArrayGetNumber (
    FPD Object objArray,
    FS DWORD index
);
```

**Description**

Gets a number with specified position.

**Parameter**

---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

index	[In] Specifies the zero-based index in the array.
-------	---

---

**Return**

A FIX24.8 number.

**Head file reference**

fpd\_objsTempl.h: 603

**FPDArrayGetRect****Syntax**

```
FS_FloatRect FPDArrayGetRect (
    FPD Object objArray
);
```

**Description**

Gets a rectangle from the array.

**Parameter**

---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

**Return**

A rectangle from the array.

**Head file reference**

fpd\_objsTempl.h: 573

**FPDArrayGetStream****Syntax**

```
FPD_Object FPDArrayGetStream (
    FPD Object objArray,
    FS DWORD index
);
```

**Description**

Gets a stream object with specified position.

**Parameter**

---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

index	[In] Specifies the zero-based index in the array.
-------	---

---

**Return**

A stream object.

**Head file reference**

fpd\_objsTempl.h: 623

**FPDArrayGetString****Syntax**

```
void FPDArrayGetString (
    FPD Object objArray,
    FS DWORD index,
    FS ByteString* outString
);
```

**Description**

Gets a string with specified position.

**Parameter**

---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

index	[In] Specifies the zero-based index in the array.
-------	---

---

outString	[Out] A byte string.
-----------	----------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 582

**FPDArrayInsertAt**

**Syntax**

```
void FPDArrayInsertAt (
    FPD Object objArray,
    FS\_DWORD index,
    FPD Object otherObj,
    void* objs
);
```

**Description**

Inserts an element at specified position.

**Parameter**

objArray	[In] The input PDF array object.
----------	----------------------------------

index	[In] Specifies the zero-based index in the array.
-------	---

otherObj	[In] The input object.
----------	------------------------

objs	[In] The indirect object collection, it can be a FDF_Document object or a FPD_Document object, required if pObj is an indirect object. In this case, a reference object will be created and inserted into the array.
------	--

**Return**

void

**Head file reference**

[fpd\\_objsTempl.h: 666](#)

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FPDArrayIsIdentical****Syntax**

```
FS_BOOL FPDArrayIsIdentical (
    FPD Object objArray,
    FPD Object otherArray
);
```

**Description**

Compares with another object.

**Parameter**

---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

otherArray	[In] The other PDF array object.
------------	----------------------------------

---

**Return**

Non-zero means identical, otherwise not identical.

**Head file reference**

fpd\_objsTempl.h: 787

**FPDArrayNew****Syntax**

```
FPD_Object FPDArrayNew (void );
```

**Description**

Creates an empty array object.

**Return**

An empty array object.

**Head file reference**

fpd\_objsTempl.h: 525

**Related method**

[FPDOBJECTDESTROY](#)

**FPDArrayRemoveAt****Syntax**

```
void FPDArrayRemoveAt (
    FPD_Object objArray,
    FS_DWORD index
);
```

**Description**

Removes an element.

**Parameter**


---

objArray	[In] The input PDF array object.
----------	----------------------------------

---

index	[In] Specifies the zero-based index in the array.
-------	---

---

**Return**

---

```
void
```

**Head file reference**

fpd\_objsTempl.h: 679

**FPDArraySetAt****Syntax**

```
void FPDArraySetAt (
    FPD Object objArray,
    FS DWORD index,
    FPD Object otherObj,
    void* objs
);
```

**Description**

Changes the element at specified position.

**Parameter**

objArray	[In] The input PDF array object.
index	[In] Specifies the zero-based index in the array.
otherObj	[In] The input object.
objs	[In] The indirect object collection, it can be a FDF_Document object or a FPD_Document object, required if pObj is an indirect object. In this case, a reference object will be created and inserted into the array.

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 653

**Since**[SDK LATEEST VERSION > 1.0](#)**FPD\_BackgroundDrawHandler**[Return from Used by](#)**Description**

PDF background drawing interface. Use for holding a [FPD\\_BackgroundDraw](#).

## Returned from

[FPDRenderContextCreateBackgroundDrawHandler](#)

## Used by

[FPDRenderContextDeleteBackgroundDrawHandler](#)  
[FPDRenderContextSetBackground](#)

## Callbacks

### Callbacks summary

#### [OnDrawBackground](#)

The callback for drawing background region.

### Callbacks detail

#### OnDrawBackground

##### Syntax

```
typedef void (*OnDrawBackground)(  
    FPD\_RenderDevice bitmapDevice,  
    FS\_AffineMatrix original2Bitmap  
>);
```

##### Description

The callback for drawing background region. For custom background PDF rendering on a non-display device (like printers), it's often not possible for rendering engine to get a background (for masked operations or transparent operations), therefore the rendering application has to provide a routine to draw the custom background into a bitmap device. In most cases, only portion of the background is needed. Default implementation fills white color.

##### Parameter

---

bitmapDevice	[In] The temporary bitmap device. The size of this device can be only portion of the rendering target device.
--------------	---

---

original2Bitmap	[In] The matrix from original device space to bitmap device space.
-----------------	--

---

##### Return

void

##### Head file reference

fpd\_renderExpT.h: 282

##### Group

### [FPD\\_BackgroundDraw](#)

**Note:** The bitmap device for rendering background might have different resolution from the original device, for example, in order to reduce spooling size, we use lower resolution for rendering shading areas and transparent images.

## **FPD\_Bookmark**

### [Return from Used by](#)

#### Description

A bookmark corresponds to an outline object in a PDF document ( *see Section 8.2.2, Document Outline, in the PDF Reference* ). Each bookmark has a title that appears on screen, and an action that specifies what happens when a user clicks on the bookmark. Bookmarks can either be created interactively by the user through the Foxit Reader user interface or programmatically generated. The typical action for a user-created bookmark is to move to another location in the current document, although any action (see FPD\_Action) can be specified. See [FPDBookmarkNew](#) , [FPDBookmarkDestroy](#) .

#### Returned from

### [FPDBookmarkNew](#)

#### Used by

[FPDBookmarkDestroy](#)  
[FPDBookmarkGetAction](#)  
[FPDBookmarkGetColorRef](#)  
[FPDBookmarkGetDest](#)  
[FPDBookmarkGetDictionary](#)  
[FPDBookmarkGetFirstChild](#)  
[FPDBookmarkGetFontStyle](#)  
[FPDBookmarkGetNextSibling](#)  
[FPDBookmarkGetTitle](#)  
[FPDBookmarkIsValid](#)

## Definitions

### Definitions summary

#### [FPD\\_BOOKMARK\\_BOLD](#)

Bold style.

#### [FPD\\_BOOKMARK\\_ITALIC](#)

Italic style.

### Definitions detail

#### **FPD\_BOOKMARK\_BOLD**

##### Syntax

```
#define FPD_BOOKMARK_BOLD 2
```

**Description**

Bold style.

**Group**

[FPDBookmarkFontFlags](#)

**Head file reference**

fpd\_docExpT.h: 287

## FPD\_BOOKMARK\_ITALIC

**Syntax**

```
#define FPD_BOOKMARK_ITALIC 1
```

**Description**

Italic style.

**Group**

[FPDBookmarkFontFlags](#)

**Head file reference**

fpd\_docExpT.h: 285

## Functions

### Functions summary

[\*\*FPDBookmarkDestroy\*\*](#)

Destroys a PDF Bookmark object created by FPDBookmarkNew.

[\*\*FPDBookmarkGetAction\*\*](#)

Gets the PDF action of a bookmark.

[\*\*FPDBookmarkGetColorRef\*\*](#)

Gets the color of a bookmark. In Windows COLORREF format: 0x00ggbbrr.

[\*\*FPDBookmarkGetDest\*\*](#)

Gets the destination of a bookmark.

[\*\*FPDBookmarkGetDictionary\*\*](#)

Gets the outline item dictionary.

[\*\*FPDBookmarkGetFirstChild\*\*](#)

Gets the first child bookmark of specified parent bookmark. If *pParent* is [NULL](#), gets top level items.

[\*\*FPDBookmarkGetFontSize\*\*](#)

Gets the font style of a bookmark. Italic and/or bold.

[\*\*FPDBookmarkGetNextSibling\*\*](#)

Gets the next sibling bookmark of specified bookmark in the same level.

[\*\*FPDBookmarkGetTitle\*\*](#)

Gets the title of a bookmark. A unicode encoded string is returned.

**FPDBookmarkIsValid**

Tests whether the bookmark is valid.

**FPDBookmarkNew**

Creates a PDF Bookmark object from a outline item dictionary.

## Functions detail

### FPDBookmarkDestroy

**Syntax**

```
void FPDBookmarkDestroy (
    FPD Bookmark bookmark
);
```

**Description**

Destroys a PDF Bookmark object created by FPDBookmarkNew.

**Parameter**

---

bookmark	[In] A PDF Bookmark object created by FPDBookmarkNew.
----------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 846

### FPDBookmarkGetAction

**Syntax**

```
void FPDBookmarkGetAction (
    FPD Bookmark bookmark,
    FPD Action* outAction
);
```

**Description**

Gets the PDF action of a bookmark.

**Parameter**

---

bookmark	[In] The input bookmark.
----------	--------------------------

---

outAction	[Out] It receives the PDF action of a bookmark.
-----------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 903

**FPDBookmarkGetColorRef****Syntax**

```
FS_DWORD FPDBookmarkGetColorRef (
    FPD\_Bookmark bookmark
);
```

**Description**

Gets the color of a bookmark. In Windows COLORREF format: 0x00ggbrr.

**Parameter**

---

bookmark	[In] The input bookmark.
----------	--------------------------

---

**Return**

The color of a bookmark.

**Head file reference**

fpd\_docTempl.h: 864

**FPDBookmarkGetDest****Syntax**

```
void FPDBookmarkGetDest (
    FPD\_Bookmark bookmark,
    FPD\_Document doc,
    FPD\_Dest* outDest
);
```

**Description**

Gets the destination of a bookmark.

**Parameter**

---

bookmark	[In] The input bookmark.
----------	--------------------------

---

doc	[In] The input PDF document.
-----	------------------------------

---

outDest	[Out] A PDF destination object.
---------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 892

**FPDBookmarkGetDictionary****Syntax**

```
FPD_Object FPDBookmarkGetDictionary (
    FPD Bookmark bookmark
);
```

**Description**

Gets the outline item dictionary.

**Parameter**

---

bookmark	[In] The input bookmark.
----------	--------------------------

---

**Return**

The outline item dictionary.

**Head file reference**

fpd\_docTempl.h: 913

**FPDBookmarkGetFirstChild****Syntax**

```
FS_BOOL FPDBookmarkGetFirstChild (
    FPD Document doc,
    FPD Bookmark parent,
    FPD Bookmark* outFirstChild
);
```

**Description**

Gets the first child bookmark of specified parent bookmark. If *pParent* is [NULL](#), gets top level items.

**Parameter**

---

doc	[In] The input PDF document.
-----	------------------------------

---

parent	[In] The input parent bookmark.
--------	---------------------------------

---

outFirstChild	[Out] The first child bookmark.
---------------	---------------------------------

---

**Return**

[TRUE](#) if the child bookmark exist,otherwise [FALSE](#) .

**Head file reference**

fpd\_docTempl.h: 922

**FPDBookmarkGetFontStyle****Syntax**

```
FS_DWORD FPDBookmarkGetFontStyle (
    FPD Bookmark bookmark
);
```

**Description**

Gets the font style of a bookmark. Italic and/or bold.

**Parameter**

---

bookmark	[In] The input bookmark.
----------	--------------------------

---

**Return**

The font style of a bookmark.

**Head file reference**

fpd\_docTempl.h: 873

**FPDBookmarkGetNextSibling****Syntax**

```
FS_BOOL FPDBookmarkGetNextSibling (
    FPD Document doc,
    FPD Bookmark bookmark,
    FPD Bookmark* outNextSibling
);
```

**Description**

Gets the next sibling bookmark of specified bookmark in the same level.

**Parameter**

---

doc	[In] The input PDF document.
-----	------------------------------

---

bookmark	[In] The input bookmark.
----------	--------------------------

---

outNextSibling	[Out] The next sibling bookmark.
----------------	----------------------------------

---

**Return**

[TRUE](#) if the next sibling bookmark exist, otherwise [FALSE](#) .

**Head file reference**

fpd\_docTempl.h: 934

**FPDBookmarkGetTitle****Syntax**

```
void FPDBookmarkGetTitle (
    FPD\_Bookmark bookmark,
    FS\_WideString* outTitle
);
```

**Description**

Gets the title of a bookmark. A unicode encoded string is returned.

**Parameter**

---

bookmark	[In] The input bookmark.
----------	--------------------------

---

outTitle	[Out] It receives the title of a bookmark.
----------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 882

**FPDBookmarkIsVaild****Syntax**

```
FS_BOOL FPDBookmarkIsVaild (
    FPD\_Bookmark bookmark
);
```

**Description**

Tests whether the bookmark is valid.

**Parameter**

---

bookmark	[In] The input bookmark.
----------	--------------------------

---

**Return**

[TRUE](#) for being valid.

**Head file reference**

fpd\_docTempl.h: 855

## FPDBookmarkNew

### Syntax

```
FPD_Bookmark FPDBookmarkNew (
    FPD Object outlineDict
);
```

### Description

Creates a PDF Bookmark object from a outline item dictionary.

### Parameter

---

outlineDict	[In] The input outline item dictionary.
-------------	---

---

### Return

A PDF Bookmark object

### Head file reference

fpd\_docTempl.h: 837

## FPD\_Boolean

### Description

FPD\_Boolean objects can have a value of [TRUE](#) or [FALSE](#).

### Functions

#### Functions summary

##### [FPDBooleanIdentical](#)

Compares with another boolean object.

##### [FPDBooleanNew](#)

Creates a boolean object from a single boolean value.

#### Functions detail

##### [FPDBooleanIdentical](#)

### Syntax

```
FS_BOOL FPDBooleanIdentical (
    FPD Object ObjBoolean,
    FPD Object otherBoolean
);
```

**Description**

Compares with another boolean object.

**Parameter**

---

ObjBoolean	[In] The boolean object.
otherBoolean	[In] The other boolean object.

---

**Return**

Non-zero means identical, otherwise not identical.

**Head file reference**

fpd\_objsTempl.h: 264

**FPDBooleanNew****Syntax**

```
FPD_Object FPDBooleanNew (
    FS_BOOL value
);
```

**Description**

Creates a boolean object from a single boolean value.

**Parameter**

---

value	[In] A boolean value.
-------	-----------------------

---

**Return**

A boolean object.

**Head file reference**

fpd\_objsTempl.h: 254

**Related method**

[FPDOObjectDestroy](#)

## FPD\_CIDFont

**Description**

The CIDFont. A CIDFont is designed to contain a large number of glyph procedures and is used for languages such as Chinese, Japanese, or Korean. Instead of being accessed by a name, each glyph procedure is accessed by an integer known as a character identifier or

CID. Instead of a font encoding, CIDFonts use a CMap to define the mapping from character codes to a font number and a character selector.

## Functions

### Functions summary

#### [\*\*FPDCIDFontAppendChar\*\*](#)

Append a charcode to a string buffer.

#### [\*\*FPDCIDFontCIDFromCharCode\*\*](#)

Gets the CID code from charcode.

#### [\*\*FPDCIDFontCountChar\*\*](#)

Gets the count of characters in a string.

#### [\*\*FPDCIDFontDestroy\*\*](#)

Destroys the CID specific font.

#### [\*\*FPDCIDFontGetCharBBox\*\*](#)

Gets the char bounding box.

#### [\*\*FPDCIDFontGetCharset\*\*](#)

Gets the character set.

#### [\*\*FPDCIDFontGetCharSize\*\*](#)

Gets the number of bytes for the char code.

#### [\*\*FPDCIDFontGetCharWidthF\*\*](#)

Gets the font char width.

#### [\*\*FPDCIDFontGetCIDTransform\*\*](#)

Gets the CID transform.

#### [\*\*FPDCIDFontGetNextChar\*\*](#)

Gets a charcode from a string at specified position.

#### [\*\*FPDCIDFontGetVertOrigin\*\*](#)

Gets character origin in vertical writing.

#### [\*\*FPDCIDFontGetVertWidth\*\*](#)

Gets glyph width in vertical writing.

#### [\*\*FPDCIDFontGlyphFromCharCode\*\*](#)

The glyph index for a charcode.

#### [\*\*FPDCIDFontIsTrueType\*\*](#)

Test whether the CID font is a True-type font.

#### [\*\*FPDCIDFontIsUnicodeCompatible\*\*](#)

Tests whether a CID font is compatible for unicode.

#### [\*\*FPDCIDFontIsVertWriting\*\*](#)

Checks whether the font is vertical writing.

#### [\*\*FPDCIDFontLoadGB2312\*\*](#)

Loads GB2312 char set.

#### [\*\*FPDCIDFontNew\*\*](#)

Creates a new empty CID specific font.

### Functions detail

#### [\*\*FPDCIDFontAppendChar\*\*](#)

##### Syntax

```
FS_INT32 FPDCIDFontAppendChar (
    FPD\_Font font,
```

```
FS_LPSTR inOutStr,  
FS_DWORD charcode  
);
```

**Description**

Append a charcode to a string buffer.

**Parameter**

font	[In] The input CID specific font.
inOutStr	[In/Out] Input a string buffer and append a charcode to it.
charcode	[In] The charcode to append.

**Return**

The number of bytes appended to the string buffer.

**Head file reference**

fpd\_resourceTempl.h: 1090

**FPDCIDFontCIDFromCharCode****Syntax**

```
FS_WORD FPDCIDFontCIDFromCharCode (  
    FPD_Font font,  
    FS_DWORD charcode  
);
```

**Description**

Gets the CID code from charcode.

**Parameter**

font	[In] The input CID specific font.
charcode	[In] Input a charcode.

**Return**

The CID of the charcode.

**Head file reference**

fpd\_resourceTempl.h: 1120

## FPDCIDFontCountChar

### Syntax

```
FS_INT32 FPDCIDFontCountChar (
    FPD_Font font,
    FS_LPCSTR str,
    FS_INT32 size
);
```

### Description

Gets the count of characters in a string.

### Parameter

---

font	[In] The input CID specific font.
------	-----------------------------------

---

str	[In] The string buffer.
-----	-------------------------

---

size	[In] The length in bytes of the string.
------	---

---

### Return

The count of characters in the string.

### Head file reference

fpd\_resourceTempl.h: 1079

## FPDCIDFontDestroy

### Syntax

```
void FPDCIDFontDestroy (
    FPD_Font font
);
```

### Description

Destroys the CID specific font.

### Parameter

---

font	[In] The input CID specific font.
------	-----------------------------------

---

### Return

void

### Head file reference

fpd\_resourceTempl.h: 1020

## FPDCIDFontGetCharBBox

### Syntax

```
FS_Rect FPDCIDFontGetCharBBox (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

### Description

Gets the char bounding box.

### Parameter

---

font	[In] The input CID specific font.
------	-----------------------------------

---

charcode	[In] Input a charcode.
----------	------------------------

---

### Return

The char bounding box.

### Head file reference

fpd\_resourceTempl.h: 1039

## FPDCIDFontGetCharset

### Syntax

```
FS_INT32 FPDCIDFontGetCharset (
    FPD\_Font font
);
```

### Description

Gets the character set.

### Parameter

---

font	[In] The input CID specific font.
------	-----------------------------------

---

### Return

The character set.

### Head file reference

fpd\_resourceTempl.h: 1139

## FPDCIDFontGetCharSize

### Syntax



```
FS_INT32 FPDCIDFontGetCharSize (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

**Description**

Gets the number of bytes for the char code.

**Parameter**

---

font	[In] The input CID specific font.
------	-----------------------------------

---

charcode	[In] Input a charcode.
----------	------------------------

---

**Return**

The number of bytes for the char code.

**Head file reference**

fpd\_resourceTempl.h: 1101

**FPDCIDFontGetCharWidthF****Syntax**

```
FS_INT32 FPDCIDFontGetCharWidthF (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

**Description**

Gets the font char width.

**Parameter**

---

font	[In] The input CID specific font.
------	-----------------------------------

---

charcode	[In] Input a charcode.
----------	------------------------

---

**Return**

The font char width.

**Head file reference**

fpd\_resourceTempl.h: 1029

**FPDCIDFontGetCIDTransform**

**Syntax**

```
FS_LPBYTE FPDCIDFontGetCIDTransform (
    FPD\_Font font,
    FS\_WORD CID
);
```

**Description**

Gets the CID transform.

**Parameter**

font	[In] The input CID specific font.
CID	[In] The input CID.

**Return**

The CID transform.

**Head file reference**

fpd\_resourceTempl.h: 1148

**FPDCIDFontGetNextChar****Syntax**

```
FS_DWORD FPDCIDFontGetNextChar (
    FPD\_Font font,
    FS\_LPCSTR str,
    FS\_INT32* inOutOffset
);
```

**Description**

Gets a charcode from a string at specified position.

**Parameter**

font	[In] The input CID specific font.
str	[In] The char buffer.
inOutOffset	[In/Out] Input the zero-based position and receive the next charcode position.

**Return**

A charcode.

**Head file reference**

---

fpd\_resourceTempl.h: 1068

### FPDCIDFontGetVertOrigin

#### Syntax

```
void FPDCIDFontGetVertOrigin (
    FPD\_Font font,
    FS\_WORD CID,
    FS\_SHORT* outVxResult,
    FS\_SHORT* outVyResult
);
```

#### Description

Gets character origin in vertical writing.

#### Parameter

---

font	[In] The input CID specific font.
------	-----------------------------------

---

CID	[In] Input a CID code.
-----	------------------------

---

outVxResult	[Out] It receives the x-coordinate of the character origin.
-------------	---

---

outVyResult	[Out] It receives the y-coordinate of the character origin.
-------------	---

---

#### Return

void

#### Head file reference

fpd\_resourceTempl.h: 1177

### FPDCIDFontGetVertWidth

#### Syntax

```
short FPDCIDFontGetVertWidth (
    FPD\_Font font,
    FS\_WORD CID
);
```

#### Description

Gets glyph width in vertical writing.

#### Parameter

---

font	[In] The input CID specific font.
------	-----------------------------------

---

---

CID	[In] Input a CID code.
-----	------------------------

---

**Return**

The glyph width in vertical writing.

**Head file reference**

fpd\_resourceTempl.h: 1167

**FPDCIDFontGlyphFromCharCode****Syntax**

```
FS_INT32 FPDCIDFontGlyphFromCharCode (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

**Description**

The glyph index for a charcode.

**Parameter**


---

font	[In] The input CID specific font.
------	-----------------------------------

---

charcode	[In] Input a charcode.
----------	------------------------

---

**Return**

The glyph index of the charcode. Return -1 for unknown.

**Head file reference**

fpd\_resourceTempl.h: 1049

**Note:** For embedded font only.

**FPDCIDFontIsTrueType****Syntax**

```
FS_BOOL FPDCIDFontIsTrueType (
    FPD\_Font font
);
```

**Description**

Test whether the CID font is a True-type font.

**Parameter**

---

font	[In] The input CID specific font.
------	-----------------------------------

---

**Return**

[TRUE](#) if the font is a True-type font.

**Head file reference**

fpd\_resourceTempl.h: 1130

**FPDCIDFontIsUnicodeCompatible****Syntax**

```
FS_BOOL FPDCIDFontIsUnicodeCompatible (
    FPD\_Font font
);
```

**Description**

Tests whether a CID font is compatible for unicode.

**Parameter**

---

font	[In] The input CID specific font.
------	-----------------------------------

---

**Return**

[TRUE](#) if the CID font is compatible for unicode, otherwise [FALSE](#).

**Head file reference**

fpd\_resourceTempl.h: 1059

**FPDCIDFontIsVertWriting****Syntax**

```
FS_BOOL FPDCIDFontIsVertWriting (
    FPD\_Font font
);
```

**Description**

Checks whether the font is vertical writing.

**Parameter**

---

font	[In] The input CID specific font.
------	-----------------------------------

---

**Return**

[TRUE](#) if the CID font is vertical writing.

**Head file reference**

fpd\_resourceTempl.h: 1158

## FPDCIDFontLoadGB2312

### Syntax

```
FS_BOOL FPDCIDFontLoadGB2312 (
    FPD_Font font
);
```

### Description

Loads GB2312 char set.

### Parameter

---

font	[In] The input CID specific font.
------	-----------------------------------

---

### Return

[TRUE](#) if the font is loaded successfully.

### Head file reference

fpd\_resourceTempl.h: 1111

## FPDCIDFontNew

### Syntax

```
FPD_Font FPDCIDFontNew (void );
```

### Description

Creates a new empty CID specific font.

### Return

A new empty CID specific font.

### Head file reference

fpd\_resourceTempl.h: 1011

# FPD\_CIDUtil

## Description

An utility class for CID processing. See [FPDCIDUtilIsVerticalJapanCID](#) .

## Functions

### Functions summary

**FPDCIDUtilIsVerticalJapanCID**

Whether the specified CID is vertical Japanese CID.

**Functions detail****FPDCIDUtilIsVerticalJapanCID****Syntax**

```
FS_BOOL FPDCIDUtilIsVerticalJapanCID (
    FS_WORD CID
);
```

**Description**

Whether the specified CID is vertical Japanese CID.

**Parameter**

CID	[In] The input CID
-----	--------------------

**Return**

Return whether the specified CID is vertical Japanese CID.

**Head file reference**

fpd\_resourceTempl.h: 1197

## FPD\_ClipPath

**Return from Used by****Description**

A clip path. See [FPDClipPathNew](#) , [FPDClipPathDestroy](#) .

**Returned from**[FPDPageObjectGetClipPath](#)[FPDClipPathNew](#)**Used by**[FPDPageArchiveLoaderLoadClipPath](#)[FPDPageArchiveSaverSaveClipPath](#)[FPDClipPathAppendPath](#)[FPDClipPathAppendTexts](#)[FPDClipPathDeletePath](#)[FPDClipPathDestroy](#)[FPDClipPathGetClipBox](#)[FPDClipPathGetClipType](#)[FPDClipPathGetModify](#)[FPDClipPathGetPath](#)

[\*\*FPDClipPathGetPathCount\*\*](#)  
[\*\*FPDClipPathGetPathPointer\*\*](#)  
[\*\*FPDClipPathGetText\*\*](#)  
[\*\*FPDClipPathGetTextCount\*\*](#)  
[\*\*FPDClipPathIsNull\*\*](#)  
[\*\*FPDClipPathSetCount\*\*](#)  
[\*\*FPDClipPathTransform\*\*](#)

## Functions

### Functions summary

#### [\*\*FPDClipPathAppendPath\*\*](#)

Appends a clipping path.

#### [\*\*FPDClipPathAppendTexts\*\*](#)

Appends clipping text objects.

#### [\*\*FPDClipPathDeletePath\*\*](#)

Removes a path from path list.

#### [\*\*FPDClipPathDestroy\*\*](#)

Destroys the clip path data object.

#### [\*\*FPDClipPathGetClipBox\*\*](#)

Gets the clip box of the clip path.

#### [\*\*FPDClipPathGetClipType\*\*](#)

Gets the clip type of specified path.

#### [\*\*FPDClipPathGetModify\*\*](#)

The interface helps init the object if the object is NULL.

#### [\*\*FPDClipPathGetPath\*\*](#)

Gets a path.

#### [\*\*FPDClipPathGetPathCount\*\*](#)

Gets the count of paths in the clip path.

#### [\*\*FPDClipPathGetPathPointer\*\*](#)

Gets the pointer to the specified path.

#### [\*\*FPDClipPathGetText\*\*](#)

Gets a text object.

#### [\*\*FPDClipPathGetTextCount\*\*](#)

Gets the count of text objects in the clip path.

#### [\*\*FPDClipPathIsNull\*\*](#)

Tests whether the path data object is [NULL](#) or not.

#### [\*\*FPDClipPathNew\*\*](#)

Creates a new empty clip path data object.

#### [\*\*FPDClipPathSetCount\*\*](#)

Estimates the count of path and text in the clip path data, and allocate the memory.

#### [\*\*FPDClipPathTransform\*\*](#)

Transforms this path.

## Functions detail

### [\*\*FPDClipPathAppendPath\*\*](#)

#### Syntax

```
void FPDClipPathAppendPath (
```

[\*\*FPD\\_ClipPath\*\*](#) path,

---

```
FPD_Path pathAppendTo,
FS_INT32 type,
FS_BOOL bAutoMerge
);
```

**Description**

Appends a clipping path.

**Parameter**

path	[In] The input clip path data object.
pathAppendTo	[In] The input clipping path.
type	[In] The clip type of the input clipping path.
bAutoMerge	[In] Whether to merge the clipping path automatically.

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 209

**FPDClipPathAppendTexts****Syntax**

```
void FPDClipPathAppendTexts (
    FPD_ClipPath path,
    FPD_PageObject* pTextsBuf,
    FS_INT32 count
);
```

**Description**

Appends clipping text objects.

**Parameter**

path	[In] The input clip path data object.
pTextsBuf	[In] Pointer to clipping text objects to append.
count	[In] The count of clipping text objects to append.

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 299

**FPDClipPathDeletePath****Syntax**

```
void FPDClipPathDeletePath (
    FPD\_ClipPath path,
    FS\_INT32 layerIndex
);
```

**Description**

Removes a path from path list.

**Parameter**

path	[In] The input clip path data object.
layerIndex	[In] The path index to remove.

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 221

**FPDClipPathDestroy****Syntax**

```
void FPDClipPathDestroy (
    FPD\_ClipPath path
);
```

**Description**

Destroys the clip path data object.

**Parameter**

path	[In] The input clip path data object.
------	---------------------------------------

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 200

**FPDClipPathGetClipBox****Syntax**

```
FS_FloatRect FPDClipPathGetClipBox (
    FPD\_ClipPath path
);
```

**Description**

Gets the clip box of the clip path.

**Parameter**

---

path	[In] The input clip path data object.
------	---------------------------------------

---

**Return**

The clip box of the clip path.

**Head file reference**

fpd\_pageobjTempl.h: 280

**FPDClipPathGetClipType****Syntax**

```
FS_INT32 FPDClipPathGetClipType (
    FPD\_ClipPath path,
    FS\_INT32 index
);
```

**Description**

Gets the clip type of specified path.

**Parameter**

---

path	[In] The input clip path data object.
------	---------------------------------------

---

index	[In] Specifies the zero-based path index in the clip path
-------	---

---

**Return**

The clip type of specified path

**Head file reference**

fpd\_pageobjTempl.h: 261

## FPDClipPathGetModify

### Syntax

```
void FPDClipPathGetModify (
    FPD\_ClipPath path
);
```

### Description

The interface helps init the object if the object is NULL.

### Parameter

---

path	[In] The input clip path data object.
------	---------------------------------------

---

### Return

void.

### Head file reference

[fpd\\_pageobjTempl.h](#): 330

### Since

[SDK LATEEST VERSION > 2.1.0.4](#)

## FPDClipPathGetPath

### Syntax

```
void FPDClipPathGetPath (
    FPD\_ClipPath path,
    FS\_INT32 index,
    FPD\_Path* outPath
);
```

### Description

Gets a path.

### Parameter

---

path	[In] The input clip path data object.
------	---------------------------------------

---

---

index	[In] Specifies the zero-based path index in the clip path.
-------	--

---

---

outPath	[Out] It receives the path.
---------	-----------------------------

---

### Return

void

**Head file reference**

fpd\_pageobjTempl.h: 231

**FPDClipPathGetPathCount****Syntax**

```
FS_DWORD FPDClipPathGetPathCount (
    FPD\_ClipPath path
);
```

**Description**

Gets the count of paths in the clip path.

**Parameter**

---

path	[In] The input clip path data object.
------	---------------------------------------

---

**Return**

The count of paths in the clip path.

**Head file reference**

fpd\_pageobjTempl.h: 231

**FPDClipPathGetPathPointer****Syntax**

```
FPD_Path FPDClipPathGetPathPointer (
    FPD\_ClipPath path,
    FS\_INT32 index
);
```

**Description**

Gets the pointer to the specified path.

**Parameter**

---

path	[In] The input clip path data object.
------	---------------------------------------

---

index	[In] Specifies the zero-based path index in the clip path.
-------	--

---

**Return**

The pointer to the specified path.

**Head file reference**

fpd\_pageobjTempl.h: 340

**Since**

## [SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

### FPDClipPathGetText

#### Syntax

```
FPD_PageObject FPDClipPathGetText (
    FPD\_ClipPath path,
    FS\_INT32 i
);
```

#### Description

Gets a text object.

#### Parameter

---

path	[In] The input clip path data object.
------	---------------------------------------

---

i	[In] Specifies the zero-based text object index in the clip path.
---	---

---

#### Return

A text object.

#### Head file reference

fpd\_pageobjTempl.h: 271

### FPDClipPathGetTextCount

#### Syntax

```
FS_DWORD FPDClipPathGetTextCount (
    FPD\_ClipPath path
);
```

#### Description

Gets the count of text objects in the clip path.

#### Parameter

---

path	[In] The input clip path data object.
------	---------------------------------------

---

#### Return

The count of text objects in the clip path.

#### Head file reference

fpd\_pageobjTempl.h: 271

### FPDClipPathIsNull

**Syntax**

```
FS_BOOL FPDClipPathIsNull (
    FPD\_ClipPath path
);
```

**Description**

Tests whether the path data object is [NULL](#) or not.

**Parameter**

---

path	[In] The input clip path data object.
------	---------------------------------------

---

**Return**

Non-zero means [NULL](#) , otherwise not [NULL](#) .

**Head file reference**

fpd\_pageobjTempl.h: 321

**FPDClipPathNew****Syntax**

```
FPD_ClipPath FPDClipPathNew (void );
```

**Description**

Creates a new empty clip path data object.

**Return**

A new empty clip path data object.

**Head file reference**

fpd\_pageobjTempl.h: 192

**FPDClipPathSetCount****Syntax**

```
void FPDClipPathSetCount (
    FPD\_ClipPath path,
    FS\_INT32 path_count,
    FS\_INT32 text_count
);
```

**Description**

Estimates the count of path and text in the clip path data, and allocate the memory.

**Parameter**

---

path	[In] The input clip path data object.
path_count	[In] The estimated count of path in the clip path data.
text_count	[In] The estimated count of text object in the clip path data.

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 310

**FPDClipPathTransform****Syntax**

```
void FPDClipPathTransform (
    FPD\_ClipPath path,
    const FS\_AffineMatrix matrix
);
```

**Description**

Transforms this path.

**Parameter**


---

path	[In] The input clip path data object.
matrix	[In] The input matrix used to transform.

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 240

**FPD\_Color****Return from Used by****Description**PDF color. see *CHAPTER 4.5 in PDF reference* . See [FPDColorNew](#) , [FPDColorDestroy](#) .**Returned from**

[FPDCColorSpaceGetBaseCS](#)  
[FPDCColorSpaceGetStockCS](#)  
[FPDCColorSpaceLoad](#)  
[FPDCColorStateGetFillColor](#)  
[FPDCColorStateGetStrokeColor](#)  
[FPDCColorStateNew](#)  
[FPDDocLoadColorSpace](#)  
[FPDPageObjectGetColorState](#)  
[FPDSHadingPatternGetColorSpace](#)  
[FPDCColorGetColorSpace](#)  
[FPDCColorGetPatternCS](#)  
[FPDCColorNew](#)

Used by

[FPDCColorSpaceCountComponents](#)  
[FPDCColorSpaceCreateBuf](#)  
[FPDCColorSpaceGetArray](#)  
[FPDCColorSpaceGetBaseCS](#)  
[FPDCColorSpaceGetBufSize](#)  
[FPDCColorSpaceGetCMYK](#)  
[FPDCColorSpaceGetDefaultColor](#)  
[FPDCColorSpaceGetDefaultValue](#)  
[FPDCColorSpaceGetFamily](#)  
[FPDCColorSpaceGetMaxIndex](#)  
[FPDCColorSpaceGetRGB](#)  
[FPDCColorSpaceReleaseCS](#)  
[FPDCColorSpaceSetCMYK](#)  
[FPDCColorSpaceSetRGB](#)  
[FPDCColorSpacesRGB](#)  
[FPDCColorSpaceTranslateImageLine](#)  
[FPDCColorStateDestroy](#)  
[FPDCColorStateGetFillColor](#)  
[FPDCColorStateGetModify](#)  
[FPDCColorStateGetStrokeColor](#)  
[FPDCColorStateIsNull](#)  
[FPDCColorStateNotUseFillColor](#)  
[FPDCColorStateSetFillColor](#)  
[FPDCColorStateSetFillPatternColor](#)  
[FPDCColorStateSetStrokeColor](#)  
[FPDCColorStateSetStrokePatternColor](#)  
[FPDFFormFindCSName](#)  
[FPDMeshStreamNew](#)  
[FPDPageFindCSName](#)  
[FPDPageArchiveLoaderLoadColorState](#)  
[FPDPageArchiveSaverSaveColorState](#)  
[FPDPageObjectSetColorState](#)  
[FPDSHadingPatternSetColorSpace](#)  
[FPDCColorCopy](#)  
[FPDCColorDestroy](#)  
[FPDCColorGetColorBuffer](#)  
[FPDCColorGetColorSpace](#)  
[FPDCColorGetPattern](#)  
[FPDCColorGetPatternColor](#)  
[FPDCColorGetPatternCS](#)  
[FPDCColorGetRGB](#)

[FPDCOLORISEQUAL](#)  
[FPDCOLORISNULL](#)  
[FPDCOLORISPATTERN](#)  
[FPDCOLORSETCOLORSPACE](#)  
[FPDCOLORSETVALUE](#)  
[FPDCOLORSETVALUE2](#)

## Functions

### Functions summary

#### [FPDCOLORCOPY](#)

Copies from another color.

#### [FPDCOLORDESTROY](#)

Destroys the PDF color object.

#### [FPDCOLORGETCOLORBUFFER](#)

Gets the component buffer.

#### [FPDCOLORGETCOLORSPACE](#)

Gets the color space

#### [FPDCOLORGETPATTERN](#)

Gets pattern information for [PDFCS\\_PATTERN](#) color space.

#### [FPDCOLORGETPATTERNCOLOR](#)

Gets component buffer for the base color space used for pattern (uncolored tiling only).

#### [FPDCOLORGETPATTERNCS](#)

Gets base color space for an uncolored tiling pattern.

#### [FPDCOLORGETRGB](#)

Converts to default RGB color space, using single byte encoding (0-255).

#### [FPDCOLORISEQUAL](#)

Compares with another color

#### [FPDCOLORISNULL](#)

Checks whether the color is [NULL](#).

#### [FPDCOLORISPATTERN](#)

Checks whether the color is a pattern.

#### [FPDCOLORNEW](#)

Creates a new empty PDF color object.

#### [FPDCOLORSETCOLORSPACE](#)

Sets the color space.

#### [FPDCOLORSETVALUE](#)

Sets color components in normal color space.

#### [FPDCOLORSETVALUE2](#)

Sets a color in pattern color space.

### Functions detail

#### [FPDCOLORCOPY](#)

##### **Syntax**

```
void FPDCOLORCOPY (
    FPD_COLOR color,
    const FPD_COLOR src
);
```

**Description**

Copies from another color.

**Parameter**

---

color	[In] The input PDF color object.
-------	----------------------------------

---

---

src
-----

---

[In] The source color object.

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 1466

**FPDColorDestroy****Syntax**

```
void FPDColorDestroy (
    FPD Color color
);
```

**Description**

Destroys the PDF color object.

**Parameter**

---

color	[In] The input PDF color object.
-------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 1429

**FPDColorGetColorBuffer****Syntax**

```
FS_FLOAT* FPDColorGetColorBuffer (
    FPD Color color
);
```

**Description**

Gets the component buffer.

**Parameter**

---

color	[In] The input PDF color object.
-------	----------------------------------

---

**Return**

[NULL](#) for unspecified color.

**Head file reference**

fpd\_resourceTempl.h: 1556

**FPDColorGetColorSpace****Syntax**

```
FPD_ColorSpace FPDColorGetColorSpace (
    FPD\_Color color
);
```

**Description**

Gets the color space

**Parameter**

---

color	[In] The input PDF color object.
-------	----------------------------------

---

**Return**

The color space

**Head file reference**

fpd\_resourceTempl.h: 1486

**FPDColorGetPattern****Syntax**

```
FPD_Pattern FPDColorGetPattern (
    FPD\_Color color
);
```

**Description**

Gets pattern information for [PDFCS\\_PATTERN](#) color space.

**Parameter**

---

color	[In] The input PDF color object.
-------	----------------------------------

---

**Return**

A pattern object. [NULL](#) if this color is not a pattern.

**Head file reference**

fpd\_resourceTempl.h: 1529

**FPDColorGetPatternColor****Syntax**

```
FS_FLOAT* FPDColorGetPatternColor (
    FPD Color color
);
```

**Description**

Gets component buffer for the base color space used for pattern (uncolored tiling only).

**Parameter**

---

color	[In] The input PDF color object.
-------	----------------------------------

---

**Return**

A buffer for the components. [NULL](#) if not applicable.

**Head file reference**

fpd\_resourceTempl.h: 1547

**FPDColorGetPatternCS****Syntax**

```
FPD_ColrSpace FPDColorGetPatternCS (
    FPD Color color
);
```

**Description**

Gets base color space for an uncolored tiling pattern.

**Parameter**

---

color	[In] The input PDF color object.
-------	----------------------------------

---

**Return**

The base color space for an uncolored tiling pattern.

**Head file reference**

fpd\_resourceTempl.h: 1538

**FPDColorGetRGB**

**Syntax**

```
FS_BOOL FPDCOLORGetRGB (
    FPD_Color color,
    FS_INT32* outR,
    FS_INT32* outG,
    FS_INT32* outB
);
```

**Description**

Converts to default RGB color space, using single byte encoding (0-255).

**Parameter**


---

color	[In] The input PDF color object.
-------	----------------------------------

---

outR	[Out] It receives the red component.
------	--------------------------------------

---

outG	[Out] It receives the green component.
------	--

---

outB	[Out] It receives the blue component.
------	---------------------------------------

---

**Return**

[TRUE](#) if the color is RGB format, otherwise [FALSE](#). If the return value is [FALSE](#), then this color should be treated as "no-color".

**Head file reference**

fpd\_resourceTempl.h: 1517

**FPDCOLORIsEqual****Syntax**

```
FS_BOOL FPDCOLORIsEqual (
    FPD_Color color,
    FPD_Color other
);
```

**Description**

Compares with another color

**Parameter**


---

color	[In] The input PDF color object.
-------	----------------------------------

---

other	[In] The another color object.
-------	--------------------------------

---

**Return**

Non-zero means equal, otherwise not equal.

**Head file reference**

fpd\_resourceTempl.h: 1447

**FPDColorIsNull****Syntax**

```
FS_BOOL FPDColorIsNull (
    FPD\_Color color
);
```

**Description**

Checks whether the color is [NULL](#).

**Parameter**

---

color	[In] The input PDF color object.
-------	----------------------------------

---

**Return**

[TRUE](#) if the color is [NULL](#).

**Head file reference**

fpd\_resourceTempl.h: 1438

**FPDColorIsPattern****Syntax**

```
FS_BOOL FPDColorIsPattern (
    FPD\_Color color
);
```

**Description**

Checks whether the color is a pattern.

**Parameter**

---

color	[In] The input PDF color object.
-------	----------------------------------

---

**Return**

[TRUE](#) if the color is a pattern, otherwise [FALSE](#).

**Head file reference**

fpd\_resourceTempl.h: 1457

## FPDColorNew

### Syntax

```
FPD_Color FPDColorNew (void );
```

### Description

Creates a new empty PDF color object.

### Return

A new PDF color object.

### Head file reference

fpd\_resourceTempl.h: 1420

## FPDColorSetColorSpace

### Syntax

```
void FPDColorSetColorSpace (
    FPD_Color color,
    FPD_ColorSpace cs
);
```

### Description

Sets the color space.

### Parameter

---

color	[In] The input PDF color object.
-------	----------------------------------

---

cs	[In] The new color space.
----	---------------------------

---

### Return

void

### Head file reference

fpd\_resourceTempl.h: 1476

## FPDColorSetValue

### Syntax

```
void FPDColorSetValue (
    FPD_Color color,
    FS_FLOAT* pCompBuf
);
```

### Description

Sets color components in normal color space.

**Parameter**

---

color	[In] The input PDF color object.
-------	----------------------------------

---

pCompBuf	[In] The input color components.
----------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 1495

**FPDColorSetValue2****Syntax**

```
void FPDColorSetValue2 (
    FPD\_Color color,
    FPD\_Pattern pattern,
    FS\_FLOAT* pCompBuf,
    FS\_INT32 ncomps
);
```

**Description**

Sets a color in pattern color space.

**Parameter**

---

color	[In] The input PDF color object.
-------	----------------------------------

---

pattern	[In] The input pattern
---------	------------------------

---

pCompBuf	[In] The components buffer.
----------	-----------------------------

---

ncomps	[In] The count of components.
--------	-------------------------------

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 1505

**FPD\_ColorSpace**

[\*\*Return from Used by\*\*](#)**Description**

The color space used in pdf color system. see *CHAPTER 4.5 in PDF reference* . See [FPDCColorSpaceLoad](#) , [FPDCColorSpaceReleaseCS](#) .

**Returned from**

[FPDCColorGetColorSpace](#)  
[FPDCColorGetPatternCS](#)  
[FPDDocLoadColorSpace](#)  
[FPDShadingPatternGetColorSpace](#)  
[FPDCColorSpaceGetBaseCS](#)  
[FPDCColorSpaceGetStockCS](#)  
[FPDCColorSpaceLoad](#)

**Used by**

[FPDCColorSetColorSpace](#)  
[FPDCColorStateSetFillColor](#)  
[FPDCColorStateSetStrokeColor](#)  
[FPDFFormFindCSName](#)  
[FPDMeshStreamNew](#)  
[FPDPPageFindCSName](#)  
[FPDShadingPatternSetColorSpace](#)  
[FPDCColorSpaceCountComponents](#)  
[FPDCColorSpaceCreateBuf](#)  
[FPDCColorSpaceGetArray](#)  
[FPDCColorSpaceGetBaseCS](#)  
[FPDCColorSpaceGetBufSize](#)  
[FPDCColorSpaceGetCMYK](#)  
[FPDCColorSpaceGetDefaultColor](#)  
[FPDCColorSpaceGetDefaultValue](#)  
[FPDCColorSpaceGetFamily](#)  
[FPDCColorSpaceGetMaxIndex](#)  
[FPDCColorSpaceGetRGB](#)  
[FPDCColorSpaceReleaseCS](#)  
[FPDCColorSpaceSetCMYK](#)  
[FPDCColorSpaceSetRGB](#)  
[FPDCColorSpacesRGB](#)  
[FPDCColorSpaceTranslateImageLine](#)

**Functions****Functions summary**[\*\*FPDCColorSpaceCountComponents\*\*](#)

Gets the number of components.

[\*\*FPDCColorSpaceCreateBuf\*\*](#)

Creates a color buffer.

[\*\*FPDCColorSpaceGetArray\*\*](#)

Gets the color space parameters array.

[\*\*FPDCColorSpaceGetBaseCS\*\*](#)

Gets based color space.

**FPDCColorSpaceGetBufSize**

Gets color buffer size.

**FPDCColorSpaceGetCMYK**

Converts a color to CMYK color space.

**FPDCColorSpaceGetDefaultColor**

Gets the default color.

**FPDCColorSpaceGetDefaultValue**

Gets the default value of a component.

**FPDCColorSpaceGetFamily**

Gets the color space family.

**FPDCColorSpaceGetMaxIndex**

Gets the max index. For Indexed color space only.

**FPDCColorSpaceGetRGB**

Converts a color to RGB color space.

**FPDCColorSpaceGetStockCS**

Gets stocked color spaces. Stocked color spaces can not be destroyed. The following color spaces are stocked: /DeviceGray, /DeviceRGB, /DeviceCMYK, /Pattern (colored patterns only).

**FPDCColorSpaceLoad**

Loads color space from a PDF object. The object can be a name or an array. The returned color space can be one of the stocked color spaces, or a new instance of one derived color space class. Application should call Release() to destroy the new instance when it's not used any more.

**FPDCColorSpaceReleaseCS**

Release the color space.

**FPDCColorSpaceSetCMYK**

Converts a color from CMYK color space.

**FPDCColorSpaceSetRGB**

Converts a color from RGB color space.

**FPDCColorSpacesRGB**

Checks whether it's sRGB or equivalent color space.

**FPDCColorSpaceTranslateImageLine**

Converts a bitmap scan line. Source must be 8bpc with default encoding, dest is be 24bpp sRGB

## Functions detail

### FPDCColorSpaceCountComponents

#### Syntax

```
FS_INT32 FPDCColorSpaceCountComponents (
    FPD_ColorSpace cs
);
```

#### Description

Gets the number of components.

#### Parameter

---

cs	[In] The input color space.
----	-----------------------------

---

**Return**

The number of components.

**Head file reference**

fpd\_resourceTempl.h: 1277

**FPDColorSpaceCreateBuf****Syntax**

```
FS_FLOAT* FPDColorSpaceCreateBuf (
    FPD\_ColorSpace cs
);
```

**Description**

Creates a color buffer.

**Parameter**

---

cs	[In] The input color space.
----	-----------------------------

---

**Return**

The created color buffer.

**Head file reference**

fpd\_resourceTempl.h: 1258

**FPDColorSpaceGetArray****Syntax**

```
FPD_Object FPDColorSpaceGetArray (
    FPD\_ColorSpace cs
);
```

**Description**

Gets the color space parameters array.

**Parameter**

---

cs	[In] The input color space.
----	-----------------------------

---

**Return**

The color space parameters array.

**Head file reference**

fpd\_resourceTempl.h: 1385

## FPDColorSpaceGetBaseCS

### Syntax

```
FPD_ColorSpace FPDColorSpaceGetBaseCS (
    FPD\_ColorSpace cs
);
```

### Description

Gets based color space.

### Parameter

---

cs	[In] The input color space.
----	-----------------------------

---

### Return

The based color space.

### Head file reference

fpd\_resourceTempl.h: 1403

## FPDColorSpaceGetBufSize

### Syntax

```
FS_INT32 FPDColorSpaceGetBufSize (
    FPD\_ColorSpace cs
);
```

### Description

Gets color buffer size.

### Parameter

---

cs	[In] The input color space.
----	-----------------------------

---

### Return

The color buffer size.

### Head file reference

fpd\_resourceTempl.h: 1249

## FPDColorSpaceGetCMYK

### Syntax

```
FS_BOOL FPDColorSpaceGetCMYK (
```

---

```
FPD_ColorSpace cs,
FS_FLOAT* pBuf,
FS_FLOAT* outC,
FS_FLOAT* outM,
FS_FLOAT* outY,
FS_FLOAT* outK
);
```

**Description**

Converts a color to CMYK color space.

**Parameter**


---

cs	[In] The input color space.
----	-----------------------------

---

pBuf	[In] The input color components.
------	----------------------------------

---

outC	[Out] It receives the C component
------	-----------------------------------

---

outM	[Out] It receives the M component
------	-----------------------------------

---

outY	[Out] It receives the Y component
------	-----------------------------------

---

outK	[Out] It receives the K component
------	-----------------------------------

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_resourceTempl.h: 1343

**FPDCOLORSPACEGETDEFAULTCOLOR****Syntax**

```
void FPDCOLORSPACEGETDEFAULTCOLOR (
    FPD_ColorSpace cs,
    FS_FLOAT* outBuf
);
```

**Description**

Gets the default color.

**Parameter**


---

cs	[In] The input color space.
----	-----------------------------

---

outBuf	[Out] It receives the color components.
--------	---

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 1267

**FPDColorSpaceGetDefaultValue****Syntax**

```
void FPDColorSpaceGetDefaultValue (
    FPD\_ColorSpace cs,
    FS\_INT32 iComponent,
    FS\_FLOAT* outValue,
    FS\_FLOAT* outMin,
    FS\_FLOAT* outMax
);
```

**Description**

Gets the default value of a component.

**Parameter**


---

cs	[In] The input color space.
----	-----------------------------

---



---

iComponent	[In] The zero-based component index.
------------	--------------------------------------

---



---

outValue	[Out] It receives the component value.
----------	--

---



---

outMin	[Out] It receives the minimize component value valid.
--------	---

---



---

outMax	[Out] It receives the maximize component value valid.
--------	---

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 1295

**FPDColorSpaceGetFamily****Syntax**

```
FS\_INT32 FPDColorSpaceGetFamily (
    FPD\_ColorSpace cs
```

);

**Description**

Gets the color space family.

**Parameter**

---

cs	[In] The input color space.
----	-----------------------------

---

**Return**

The color space family.

**Head file reference**

fpd\_resourceTempl.h: 1286

**FPDColorSpaceGetMaxIndex****Syntax**

```
FS_INT32 FPDColorSpaceGetMaxIndex (
    FPD\_ColorSpace cs
);
```

**Description**

Gets the max index. For Indexed color space only.

**Parameter**

---

cs	[In] The input color space.
----	-----------------------------

---

**Return**

The max index. For Indexed color space only.

**Head file reference**

fpd\_resourceTempl.h: 1394

**FPDColorSpaceGetRGB****Syntax**

```
FS_BOOL FPDColorSpaceGetRGB (
    FPD\_ColorSpace cs,
    FS\_FLOAT* pBuf,
    FS\_FLOAT* outR,
    FS\_FLOAT* outG,
    FS\_FLOAT* outB
);
```

**Description**

Converts a color to RGB color space.

**Parameter**

cs	[In] The input color space.
pBuf	[In] The input color components.
outR	[Out] It receives the red component.
outG	[Out] It receives the green component.
outB	[Out] It receives the blue component.

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_resourceTempl.h: 1317

**FPDCColorSpaceGetStockCS****Syntax**

```
FPD_ColorSpace FPDCColorSpaceGetStockCS (
    FS_INT32 family
);
```

**Description**

Gets stocked color spaces. Stocked color spaces can not be destroyed. The following color spaces are stocked: /DeviceGray, /DeviceRGB, /DeviceCMYK, /Pattern (colored patterns only).

**Parameter**

family	[In] The color space family. See <a href="#">FPDFFontColorSpaceFamilies</a> .
--------	---

**Return**

The stocked color spaces.

**Head file reference**

fpd\_resourceTempl.h: 1214

**FPDCColorSpaceLoad**

**Syntax**

```
FPD_ColorSpace FPDCOLORSPACELoad (
    FPD Document doc,
    FPD Object CSObj
);
```

**Description**

Loads color space from a PDF object. The object can be a name or an array. The returned color space can be one of the stocked color spaces, or a new instance of one derived color space class. Application should call Release() to destroy the new instance when it's not used any more.

**Parameter**

---

doc	[In] The PDF document.
-----	------------------------

---

CSObj	[In] The PDF object of a color space.
-------	---------------------------------------

---

**Return**

The color space loaded from a PDF object.

**Head file reference**

fpd\_resourceTempl.h: 1225

**FPDCOLORSPACEReleaseCS****Syntax**

```
void FPDCOLORSPACEReleaseCS (
    FPD ColorSpace cs
);
```

**Description**

Release the color space.

**Parameter**

---

cs	[In] The input color space.
----	-----------------------------

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 1239

**Note:** Applications should not delete color space directly, because some color spaces are stocked. Call Release() function instead.

## FPDColorSpaceSetCMYK

### Syntax

```
FS_BOOL FPDColorSpaceSetCMYK (
    FPD_ColorSpace cs,
    FS_FLOAT* outBuf,
    FS_FLOAT c,
    FS_FLOAT m,
    FS_FLOAT y,
    FS_FLOAT k
);
```

### Description

Converts a color from CMYK color space.

### Parameter

cs	[In] The input color space.
outBuf	[Out] It receives the converted color.
c	[In] The C component of the color.
m	[In] The M component of the color.
y	[In] The Y component of the color.
k	[In] The K component of the color.

### Return

Non-zero means success, otherwise failure.

### Head file reference

fpd\_resourceTempl.h: 1357

## FPDColorSpaceSetRGB

### Syntax

```
FS_BOOL FPDColorSpaceSetRGB (
    FPD_ColorSpace cs,
    FS_FLOAT* outBuf,
    FS_FLOAT R,
    FS_FLOAT G,
    FS_FLOAT B
);
```

**Description**

Converts a color from RGB color space.

**Parameter**


---

cs	[In] The input color space.
outBuf	[Out] It receives the converted color.
R	[In] The red component of the color.
G	[In] The green component of the color.
B	[In] The blue component of the color.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_resourceTempl.h: 1330

**FPDColorSpacesRGB****Syntax**

```
FS_BOOL FPDColorSpacesRGB (
    FPD\_ColorSpace cs
);
```

**Description**

Checks whether it's sRGB or equivalent color space.

**Parameter**


---

cs	[In] The input color space.
----	-----------------------------

---

**Return**

[TRUE](#) if the *cs* is sRGB or equivalent color space.

**Head file reference**

fpd\_resourceTempl.h: 1308

**FPDColorSpaceTranslateImageLine****Syntax**

```
void FPDColorSpaceTranslateImageLine (
```

---

```
FPD_ColorSpace cs,
FS_LPBYTE destBuf,
FS_LPCBYTE srcBuf,
FS_INT32 pixels,
FS_INT32 imageWidth,
FS_INT32 imageHeight
);
```

**Description**

Converts a bitmap scan line. Source must be 8bpc with default encoding, dest is be 24bpp sRGB

**Parameter**


---

cs	[In] The input color space.
----	-----------------------------

---

destBuf	[In] The destinate line buffer.
---------	---------------------------------

---

srcBuf	[In] The source line buffer.
--------	------------------------------

---

pixels	[In] The pixel in the line.
--------	-----------------------------

---

imageWidth	[In] The image_width.
------------	-----------------------

---

imageHeight	[In] The image height.
-------------	------------------------

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 1371

## FPD\_ColorState

**Return from Used by****Description**

The color state for rendering. See [FPDColorStateNew](#) , [FPDColorStateDestroy](#) .

**Returned from**

[FPDPageObjectGetColorState](#)  
[FPDColorStateNew](#)

**Used by**

[FPDPageArchiveLoaderLoadColorState](#)  
[FPDPageArchiveSaverSaveColorState](#)  
[FPDPageObjectSetColorState](#)  
[FPDCOLORSTATEDESTROY](#)  
[FPDCOLORSTATEGETFILLCOLOR](#)  
[FPDCOLORSTATEGETMODIFY](#)  
[FPDCOLORSTATEGETSTROKECOLOR](#)  
[FPDCOLORSTATEISNULL](#)  
[FPDCOLORSTATENOTUSEFILLCOLOR](#)  
[FPDCOLORSTATESETFILLCOLOR](#)  
[FPDCOLORSTATESETFILLPATTERNCOLOR](#)  
[FPDCOLORSTATESETSTROKECOLOR](#)  
[FPDCOLORSTATESETSTROKEPATTERNCOLOR](#)

## Functions

### Functions summary

#### [FPDCOLORSTATEDESTROY](#)

Destroys the PDF color state object.

#### [FPDCOLORSTATEGETFILLCOLOR](#)

Gets the filling color.

#### [FPDCOLORSTATEGETMODIFY](#)

The interface helps init the object if the object is NULL.

#### [FPDCOLORSTATEGETSTROKECOLOR](#)

Gets the stroking color.

#### [FPDCOLORSTATEISNULL](#)

Tests whether the color state object is [NULL](#) or not.

#### [FPDCOLORSTATENEW](#)

Creates a new empty PDF color state object.

#### [FPDCOLORSTATENOTUSEFILLCOLOR](#)

Do not use the filling mode.

#### [FPDCOLORSTATESETFILLCOLOR](#)

Sets the filling normal color.

#### [FPDCOLORSTATESETFILLPATTERNCOLOR](#)

Sets the filling pattern color.

#### [FPDCOLORSTATESETSTROKECOLOR](#)

Sets the stroking normal color.

#### [FPDCOLORSTATESETSTROKEPATTERNCOLOR](#)

Sets the stroking pattern color.

## Functions detail

### FPDCOLORSTATEDESTROY

#### Syntax

```
void FPDCOLORSTATEDESTROY (
    FPD_COLORSTATE clrState
);
```

#### Description

Destroys the PDF color state object.

**Parameter**

---

clrState	[In] The input PDF color state object.
----------	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 368

**FPDColorStateGetFillColor****Syntax**

```
FPD_Color FPDColorStateGetFillColor (
    FPD\_ColorState clrState
);
```

**Description**

Gets the filling color.

**Parameter**

---

clrState	[In] The input PDF color state object.
----------	--

---

**Return**

The filling color.

**Head file reference**

fpd\_pageobjTempl.h: 377

**FPDColorStateGetModify****Syntax**

```
void FPDColorStateGetModify (
    FPD\_ColorState clrState
);
```

**Description**

The interface helps init the object if the object is NULL.

**Parameter**

---

clrState	[In] The input PDF color state object.
----------	--

---

**Return**

void.

**Head file reference**

fpd\_pageobjTempl.h: 452

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FPDColorStateGetStrokeColor****Syntax**

```
FPD_Color FPDColorStateGetStrokeColor (
    FPD\_ColorState clrState
);
```

**Description**

Gets the stroking color.

**Parameter**

---

clrState	[In] The input PDF color state object.
----------	--

---

**Return**

The stroking color.

**Head file reference**

fpd\_pageobjTempl.h: 386

**FPDColorStateIsNull****Syntax**

```
FS_BOOL FPDColorStateIsNull (
    FPD\_ColorState clrState
);
```

**Description**

Tests whether the color state object is [NULL](#) or not.

**Parameter**

---

clrState	[In] The input PDF color state object.
----------	--

---

**Return**

Non-zero means [NULL](#), otherwise not [NULL](#).

**Head file reference**

fpd\_pageobjTempl.h: 443

## FPDColorStateNew

### Syntax

```
FPD_ColorState FPDColorStateNew (void );
```

### Description

Creates a new empty PDF color state object.

### Return

A new empty PDF color state object.

### Head file reference

fpd\_pageobjTempl.h: 359

## FPDColorStateNotUseFillColor

### Syntax

```
void FPDColorStateNotUseFillColor (
    FPD\_ColorState clrState
);
```

### Description

Do not use the filling mode.

### Parameter

---

clrState	[In] The input PDF color state object.
----------	--

---

### Return

void

### Head file reference

fpd\_pageobjTempl.h: 462

### Since

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FPDColorStateSetFillColor

### Syntax

```
void FPDColorStateSetFillColor (
    FPD\_ColorState clrState,
    FPD\_ColorSpace clrSpace,
    FS\_FLOAT* pValue,
    FS\_INT32 nValues
);
```

**Description**

Sets the filling normal color.

**Parameter**

clrState	[In] The input PDF color state object.
clrSpace	[In] The color space of the filling color.
pValue	[In] The color component values in the specified color space.
nValues	[In] The count of the input parameters.

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 395

**FPDColorStateSetFillPatternColor****Syntax**

```
void FPDColorStateSetFillPatternColor (
    FPD_ColorState clrState,
    FPD_Pattern pattern,
    FS_FLOAT* pValue,
    int nValues
);
```

**Description**

Sets the filling pattern color.

**Parameter**

clrState	[In] The input PDF color state object.
pattern	[In] The input pattern.
pValue	[In] The input parameters for the pattern color.
nValues	[In] The count of the input parameters.

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 419

**FPDColorStateSetStrokeColor****Syntax**

```
void FPDColorStateSetStrokeColor (
    FPD\_ColorState clrState,
    FPD\_ColorSpace clrSpace,
    FS\_FLOAT* pValue,
    FS\_INT32 nValues
);
```

**Description**

Sets the stroking normal color.

**Parameter**

---

clrState	[In] The input PDF color state object.
clrSpace	[In] The color space of the stroking color.
pValue	[In] The color component values in the specified color space.
nValues	[In] The count of the input parameters.

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 407

**FPDColorStateSetStrokePatternColor****Syntax**

```
void FPDColorStateSetStrokePatternColor (
    FPD\_ColorState clrState,
    FPD\_Pattern pattern,
    FS\_FLOAT* pValue,
    int nValues
);
```

**Description**

Sets the stroking pattern color.

**Parameter**

clrState	[In] The input PDF color state object.
pattern	[In] The input pattern.
pValue	[In] The input parameters for the pattern color.
nValues	[In] The count of the input parameters.

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 431

## FPD\_ConnectedInfo

**[Return from Used by](#)****Description**

The connected PDF info object can get connected PDF document ID and version ID etc.  
Typical usage of this object is to get and set connected PDF info.

**Returned from****[FPDConnectedInfoNew](#)****Used by**

**[FPDConnectedInfoCheckSettingOpenAction](#)**  
**[FPDConnectedInfoDeleteOpenAction](#)**  
**[FPDConnectedInfoDestroy](#)**  
**[FPDConnectedInfoGetEncryptEnvelope](#)**  
**[FPDConnectedInfoGetEncryptOffline](#)**  
**[FPDConnectedInfoGetEndpoint](#)**  
**[FPDConnectedInfoGetId](#)**  
**[FPDConnectedInfoGetTracking](#)**  
**[FPDConnectedInfoIsConnectedPDF](#)**  
**[FPDConnectedInfoSetEncryptEnvelope](#)**  
**[FPDConnectedInfoSetEncryptOffline](#)**  
**[FPDConnectedInfoSetEndpoint](#)**  
**[FPDConnectedInfoSetId](#)**  
**[FPDConnectedInfoSetOpenActionURL](#)**  
**[FPDConnectedInfoSetOpenActionURL2](#)**  
**[FPDConnectedInfoSetTracking](#)**

## [FPDConnectedInfoUpdate](#)

### Definitions

#### Definitions summary

##### [FPD\\_CONNECT\\_DOCUMENTID](#)

It is document Id.

##### [FPD\\_CONNECT REVIEWID](#)

It is review Id.

##### [FPD\\_CONNECT\\_VERSIONID](#)

It is version Id.

##### [FPD\\_CONNECT\\_BOOKMARKS](#)

Tracking bookmarks.

##### [FPD\\_CONNECT\\_COMMENT](#)

Tracking commenting.

##### [FPD\\_CONNECT\\_COPY](#)

Tracking copying.

##### [FPD\\_CONNECT\\_EXTRACT](#)

Tracking extracting.

##### [FPD\\_CONNECT\\_FORMFILL](#)

Tracking filling form.

##### [FPD\\_CONNECT\\_PAGES](#)

Tracking pages.

##### [FPD\\_CONNECT\\_PRINTING](#)

Tracking printing.

##### [FPD\\_CONNECT\\_SIGN](#)

Tracking signing.

##### [FPD\\_CONNECT\\_SUBMIT\\_RECEIVE](#)

Receives open message.

##### [FPD\\_CONNECT\\_SUBMIT\\_UPDATE](#)

Update open message.

### Definitions detail

#### [FPD\\_CONNECT\\_DOCUMENTID](#)

##### **Syntax**

```
#define FPD_CONNECT_DOCUMENTID 0x01
```

##### **Description**

It is document Id.

##### **Group**

[FPDConnectedInfoIdTypes](#)

##### **Head file reference**

fpd\_docExpT.h: 1351

## FPD\_CONNECT\_REVIEWID

**Syntax**

```
#define FPD_CONNECT_REVIEWID 0x03
```

**Description**

It is review Id.

**Group**

[FPDConnectedInfoIdTypes](#)

**Head file reference**

fpd\_docExpT.h: 1357

## FPD\_CONNECT\_VERSIONID

**Syntax**

```
#define FPD_CONNECT_VERSIONID 0x02
```

**Description**

It is version Id.

**Group**

[FPDConnectedInfoIdTypes](#)

**Head file reference**

fpd\_docExpT.h: 1354

## FPD\_CONNECT\_BOOKMARKS

**Syntax**

```
#define FPD_CONNECT_BOOKMARKS 0x0010
```

**Description**

Tracking bookmarks.

**Group**

[FPDConnectedInfoTrackingTypes](#)

**Head file reference**

fpd\_docExpT.h: 1379

## FPD\_CONNECT\_COMMENT

**Syntax**

```
#define FPD_CONNECT_COMMENT 0x0004
```

**Description**

Tracking commenting.

**Group**

[FPDConnectedInfoTrackingTypes](#)

**Head file reference**

fpd\_docExpT.h: 1373

## FPD\_CONNECT\_COPY

**Syntax**

```
#define FPD_CONNECT_COPY 0x0020
```

**Description**

Tracking copying.

**Group**

[FPDConnectedInfoTrackingTypes](#)

**Head file reference**

fpd\_docExpT.h: 1382

## FPD\_CONNECT\_EXTRACT

**Syntax**

```
#define FPD_CONNECT_EXTRACT 0x0080
```

**Description**

Tracking extracting.

**Group**

[FPDConnectedInfoTrackingTypes](#)

**Head file reference**

fpd\_docExpT.h: 1388

## FPD\_CONNECT\_FORMFILL

**Syntax**

```
#define FPD_CONNECT_FORMFILL 0x0002
```

**Description**

Tracking filling form.

**Group**

[FPDConnectedInfoTrackingTypes](#)**Head file reference**

fpd\_docExpT.h: 1370

**FPD\_CONNECT\_PAGES****Syntax**

#define FPD\_CONNECT\_PAGES 0x0008

**Description**

Tracking pages.

**Group**[FPDConnectedInfoTrackingTypes](#)**Head file reference**

fpd\_docExpT.h: 1376

**FPD\_CONNECT\_PRINTING****Syntax**

#define FPD\_CONNECT\_PRINTING 0x0001

**Description**

Tracking printing.

**Group**[FPDConnectedInfoTrackingTypes](#)**Head file reference**

fpd\_docExpT.h: 1367

**FPD\_CONNECT\_SIGN****Syntax**

#define FPD\_CONNECT\_SIGN 0x0040

**Description**

Tracking signing.

**Group**[FPDConnectedInfoTrackingTypes](#)**Head file reference**

fpd\_docExpT.h: 1385

## FPD\_CONNECT\_SUBMIT\_RECEIVE

### Syntax

```
#define FPD_CONNECT_SUBMIT_RECEIVE 0x02
```

### Description

Receives open message.

### Group

[FPDConnectedOpenActionURLTypes](#)

### Head file reference

fpd\_docExpT.h: 1400

## FPD\_CONNECT\_SUBMIT\_UPDATE

### Syntax

```
#define FPD_CONNECT_SUBMIT_UPDATE 0x01
```

### Description

Update open message.

### Group

[FPDConnectedOpenActionURLTypes](#)

### Head file reference

fpd\_docExpT.h: 1398

## Functions

### Functions summary

#### [FPDConnectedInfoCheckSettingOpenAction](#)

Checks whether the OpenAction javascript server URL is set or not.

#### [FPDConnectedInfoDeleteOpenAction](#)

Deletes ConnectedPDF info in OpenAction.

#### [FPDConnectedInfoDestroy](#)

Destroys the connected PDF info object.

#### [FPDConnectedInfoGetEncryptEnvelope](#)

Gets the offline envelope.

#### [FPDConnectedInfoGetEncryptOffline](#)

Sets envelope string to encrypt dictionary.

#### [FPDConnectedInfoGetEndpoint](#)

Gets the URL of connected PDF web services.

#### [FPDConnectedInfoGetId](#)

Gets the Id of the connected PDF info object.

#### [FPDConnectedInfoGetTracking](#)

Gets the tracking type. This interface is reserved now.

**FPDConnectedInfoIsConnectedPDF**

Checks whether the PDF document is a connected PDF or not.

**FPDConnectedInfoIsConnectedPDF2**

Checks whether the PDF document is a connected PDF or not.

**FPDConnectedInfoNew**

Creates the connected PDF info object.

**FPDConnectedInfoSetEncryptEnvelope**

Sets envelope string to encrypt dictionary.

**FPDConnectedInfoSetEncryptOffline**

Sets whether the encrypted connected PDF is offline or not.

**FPDConnectedInfoSetEndpoint**

Sets the URL of connected PDF web services.

**FPDConnectedInfoSetId**

Sets the Id of the connected PDF info object.

**FPDConnectedInfoSetOpenActionURL**

Sets OpenAction javascript server URL.

**FPDConnectedInfoSetOpenActionURL2**

Sets OpenAction javascript server URL.

**FPDConnectedInfoSetTracking**

Sets the tracking type. This interface is reserved now.

**FPDConnectedInfoUpdate**

Updates the connected PDF info.

## Functions detail

### FPDConnectedInfoCheckSettingOpenAction

#### Syntax

```
FS_BOOL FPDConnectedInfoCheckSettingOpenAction (
    FPD_ConnectedInfo connectedInfo,
    FS_INT32 nType,
    FS_LPCSTR lpsURL
);
```

#### Description

Checks whether the OpenAction javascript server URL is set or not.

#### Parameter

connectedInfo	[In] The input connected PDF info object.
---------------	---

nType	[In] Sees <a href="#">FPDConnectedOpenActionURLTypes</a> definitions.
-------	---

lpsURL	[In] The input sever URL.
--------	---------------------------

#### Return

TRUE if the OpenAction javascript server URL is set.

#### Head file reference

fpd\_docTempl.h: 6542

**Since**

[SDK\\_LATEEST\\_VERSION > 8.0.2](#)

## FPDConnectedInfoDeleteOpenAction

**Syntax**

```
void FPDConnectedInfoDeleteOpenAction (
    FPD\_ConnectedInfo connectedInfo
);
```

**Description**

Deletes ConnectedPDF info in OpenAction.

**Parameter**

---

connectedInfo	[In] The input connected PDF info object.
---------------	---

---

**Return**

void.

**Head file reference**

fpd\_docTempl.h: 6554

**Since**

[SDK\\_LATEEST\\_VERSION > 8.2.1](#)

## FPDConnectedInfoDestroy

**Syntax**

```
void FPDConnectedInfoDestroy (
    FPD\_ConnectedInfo connectedInfo
);
```

**Description**

Destroys the connected PDF info object.

**Parameter**

---

connectedInfo	[In] The input connected PDF info object.
---------------	---

---

**Return**

void.

**Head file reference**

fpd\_docTempl.h: 6386

**Since**  
[SDK LATEEST VERSION > 7.1](#)

## FPDConnectedInfoGetEncryptEnvelope

### Syntax

```
FS_BOOL FPDConnectedInfoGetEncryptEnvelope (
    FPD\_ConnectedInfo connectedInfo,
    FS\_ByteString* outEnvelope
);
```

### Description

Gets the offline envelope.

### Parameter

---

connectedInfo	[In] The input connected PDF info object.
---------------	---

---

outEnvelope	[Out] It receives the offline envelope.
-------------	---

---

### Return

TRUE for success, otherwise failure.

### Head file reference

fpd\_docTempl.h: 6508

**Since**

[SDK LATEEST VERSION > 7.3.1](#)

## FPDConnectedInfoGetEncryptOffline

### Syntax

```
FS_BOOL FPDConnectedInfoGetEncryptOffline (
    FPD\_ConnectedInfo connectedInfo,
    FS\_BOOL* outIsOffline
);
```

### Description

Sets envelope string to encrypt dictionary.

### Parameter

---

connectedInfo	[In] The input connected PDF info object.
---------------	---

---

---

outIsOffline	[Out] It receives the result whether the encrypted connected PDF is offline or not.
--------------	---

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 6497

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

**FPDConnectedInfoGetEndpoint****Syntax**

```
FS_BOOL FPDConnectedInfoGetEndpoint (
    FPD\_ConnectedInfo connectedInfo,
    FS\_ByteString* outEndPoint
);
```

**Description**

Gets the URL of connected PDF web services.

**Parameter**


---

connectedInfo	[In] The input connected PDF info object.
---------------	---

---

outEndPoint	[Out] It receives the URL of connected PDF web services.
-------------	--

---

**Return**

TRUE if success, FALSE if failed.

**Head file reference**

fpd\_docTempl.h: 6575

**Since**

[SDK\\_LATEEST\\_VERSION > 8.3](#)

**FPDConnectedInfoGetId****Syntax**

```
FS_BOOL FPDConnectedInfoGetId (
    FPD\_ConnectedInfo connectedInfo,
    FS\_INT32 nIdType,
    FS\_ByteString* outURI,
    FS\_ByteString* outCert
);
```

**Description**

Gets the Id of the connected PDF info object.

**Parameter**


---

connectedInfo	[In] The input connected PDF info object.
nIdType	[In] The input Id type.
outURI	[Out] It receives the URI.
outCert	[Out] It receives the certificate. This param is reserved now.

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 6409

**Since**

[SDK\\_LATEEST\\_VERSION > 7.1](#)

**FPDConnectedInfoGetTracking****Syntax**

```
FS_BOOL FPDConnectedInfoGetTracking (
    FPD\_ConnectedInfo connectedInfo,
    FS\_INT32* outTrack
);
```

**Description**

Gets the tracking type. This interface is reserved now.

**Parameter**


---

connectedInfo	[In] The input connected PDF info object.
outTrack	[Out] It receives the tracking type.

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 6453

**Since**

[SDK LATEST VERSION > 7.1](#)

**FPDConnectedInfoIsConnectedPDF****Syntax**

```
FS_BOOL FPDConnectedInfoIsConnectedPDF (
    FPD ConnectedInfo connectedInfo
);
```

**Description**

Checks whether the PDF document is a connected PDF or not.

**Parameter**

---

connectedInfo	[In] The input connected PDF info object.
---------------	---

---

**Return**

TRUE if the PDF document is a connected PDF, otherwise not.

**Head file reference**

fpd\_docTempl.h: 6422

**Since**

[SDK LATEST VERSION > 7.1](#)

**FPDConnectedInfoIsConnectedPDF2****Syntax**

```
FS_BOOL FPDConnectedInfoIsConnectedPDF2 (
    FPD Document fpdDoc
);
```

**Description**

Checks whether the PDF document is a connected PDF or not.

**Parameter**

---

fpdDoc	[In] The input PDF document.
--------	------------------------------

---

**Return**

TRUE if the PDF document is a connected PDF, otherwise not.

**Head file reference**

fpd\_docTempl.h: 6432

**Since**  
[SDK LATEEST VERSION > 7.1](#)

### FPDConnectedInfoNew

**Syntax**

```
FPD_ConnectedInfo FPDConnectedInfoNew (
    FPD Document fpdDoc
);
```

**Description**

Creates the connected PDF info object.

**Parameter**

---

fpdDoc	[In] The input document.
--------	--------------------------

---

**Return**

The connected PDF info object.

**Head file reference**

fpd\_docTempl.h: 6376

**Since**

[SDK LATEEST VERSION > 7.1](#)

### FPDConnectedInfoSetEncryptEnvelope

**Syntax**

```
void FPDConnectedInfoSetEncryptEnvelope (
    FPD ConnectedInfo connectedInfo,
    FS ByteString bsEnvelope
);
```

**Description**

Sets envelope string to encrypt dictionary.

**Parameter**

---

connectedInfo	[In] The input connected PDF info object.
---------------	---

---

bsEnvelope	[In] The input offline envelope.
------------	----------------------------------

---

**Return**

void.

**Head file reference**

fpd\_docTempl.h: 6486

**Since**[SDK LATEEST VERSION > 7.3.1](#)**FPDConnectedInfoSetEncryptOffline****Syntax**

```
void FPDConnectedInfoSetEncryptOffline (
    FPD\_ConnectedInfo connectedInfo,
    FS\_BOOL bOffline
);
```

**Description**

Sets whether the encrypted connected PDF is offline or not.

**Parameter**

---

connectedInfo	[In] The input connected PDF info object.
---------------	---

---

bOffline	[In] The input offline flag parameter.
----------	--

---

**Return**

void.

**Head file reference**

fpd\_docTempl.h: 6475

**Since**[SDK LATEEST VERSION > 7.3.1](#)**FPDConnectedInfoSetEndpoint****Syntax**

```
void FPDConnectedInfoSetEndpoint (
    FPD\_ConnectedInfo connectedInfo,
    FS\_LPCSTR lpsEndPoint
);
```

**Description**

Sets the URL of connected PDF web services.

**Parameter**

---

connectedInfo	[In] The input connected PDF info object.
---------------	---

---

lpsEndPoint	[In] The input URL of connected PDF web services.
-------------	---

---

**Return**

void.

**Head file reference**

fpd\_docTempl.h: 6564

**Since**[SDK\\_LATEEST\\_VERSION > 8.3](#)**FPDConnectedInfoSetId****Syntax**

```
void FPDConnectedInfoSetId (
    FPD\_ConnectedInfo connectedInfo,
    FS\_INT32 nIdType,
    FS\_ByteString bsURI,
    FS\_ByteString bsCert
);
```

**Description**

Sets the Id of the connected PDF info object.

**Parameter**


---

connectedInfo	[In] The input connected PDF info object.
---------------	---

---

nIdType	[In] The input Id type.
---------	-------------------------

---

bsURI	[In] The input URI.
-------	---------------------

---

bsCert	[In] The input certificate. This param is reserved now.
--------	---

---

**Return**

void.

**Head file reference**

fpd\_docTempl.h: 6396

**Since**[SDK\\_LATEEST\\_VERSION > 7.1](#)

## FPDConnectedInfoSetOpenActionURL

### Syntax

```
FS_BOOL FPDConnectedInfoSetOpenActionURL (  
    FPD\_ConnectedInfo connectedInfo,  
    FS\_LPCSTR lpsURL  
) ;
```

### Description

Sets OpenAction javascript server URL.

### Parameter

---

connectedInfo	[In] The input connected PDF info object.
---------------	---

---

lpsURL	[In] The input sever URL.
--------	---------------------------

---

### Return

TRUE for success, otherwise failure.

### Head file reference

fpd\_docTempl.h: 6519

### Since

[SDK\\_LATEEST\\_VERSION > 7.3.1](#)

## FPDConnectedInfoSetOpenActionURL2

### Syntax

```
FS_BOOL FPDConnectedInfoSetOpenActionURL2 (  
    FPD\_ConnectedInfo connectedInfo,  
    FS\_INT32 nType,  
    FS\_LPCSTR lpsURL  
) ;
```

### Description

Sets OpenAction javascript server URL.

### Parameter

---

connectedInfo	[In] The input connected PDF info object.
---------------	---

---

nType	[In] Sees <a href="#">FPDConnectedOpenActionURLTypes</a> definitions.
-------	---

---

lpsURL	[In] The input sever URL.
--------	---------------------------

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 6530

**Since**

[SDK LATEEST VERSION > 8.0.2](#)

## FPDConnectedInfoSetTracking

**Syntax**

```
void FPDConnectedInfoSetTracking (
    FPD\_ConnectedInfo connectedInfo,
    FS\_INT32 track
);
```

**Description**

Sets the tracking type. This interface is reserved now.

**Parameter**

---

connectedInfo	[In] The input connected PDF info object.
---------------	---

---

track	[In] The input tracking type.
-------	-------------------------------

---

**Return**

void.

**Head file reference**

fpd\_docTempl.h: 6442

**Since**

[SDK LATEEST VERSION > 7.1](#)

## FPDConnectedInfoUpdate

**Syntax**

```
void FPDConnectedInfoUpdate (
    FPD\_ConnectedInfo connectedInfo,
    FS\_BOOL bOpenAction
);
```

**Description**

Updates the connected PDF info.

**Parameter**

---

connectedInfo	[In] The input connected PDF info object.
bOpenAction	[In] Whether to add open action to ConnectedPDF document or not. Sets it FALSE as default.

---

**Return**

void.

**Head file reference**

fpd\_docTempl.h: 6464

**Since**[SDK LATEEST VERSION > 7.1](#)

## FPD\_ContentGenerator

**[Return from Used by](#)****Description**

PDF Page or Form content generator.

**Returned from****[FPDFFormNewContentGenerator](#)**  
**[FPDPPageNewContentGenerator](#)****Used by****[FPDFFormContinueGenerateContent](#)**  
**[FPDFFormDestroyContentGenerator](#)**  
**[FPDFFormStartGenerateContent](#)**  
**[FPDPPageContinueGenerateContent](#)**  
**[FPDPPageDestroyContentGenerator](#)**  
**[FPDPPageStartGenerateContent](#)****Enumerations****Enumerations summary****[FPD\\_ProgressiveStatus](#)**

The status of generating PDF page or Form content progressively.

**Enumerations detail****FPD\_ProgressiveStatus****Syntax**

enum FPD\_ProgressiveStatus{

```
    FPD\_Ready,  
    FPD\_ToBeContinued,  
    FPD\_Found,  
    FPD\_NotFound,  
    FPD\_Failed,  
    FPD\_Done  
};
```

#### Description

The status of generating PDF page or Form content progressively.

#### Head file reference

fpd\_pageExpT.h: 125

##### **FPD\_Ready**

Ready.

##### **FPD\_ToBeContinued**

To be continued.

##### **FPD\_Found**

Found.

##### **FPD\_NotFound**

Not found.

##### **FPD\_Failed**

Failed.

##### **FPD\_Done**

Done.

## FPD\_ContentMark

#### [Return from Used by](#)

#### Description

The content marks for tagged pdf. See [FPDContentMarkNew](#) , [FPDContentMarkDestroy](#) .

#### Returned from

[FPDContentMarkItemNew](#)  
[FPDPPageObjectGetContentMark](#)  
[FPDPPageObjectGetContentMark2](#)  
[FPDContentMarkGetItem](#)  
[FPDContentMarkNew](#)

#### Used by

[FPDContentMarkItemDestroy](#)

[\*\*FPDContentMarkItemGetName\*\*](#)  
[\*\*FPDContentMarkItemGetParam\*\*](#)  
[\*\*FPDContentMarkItemGetParamType\*\*](#)  
[\*\*FPDContentMarkItemSetName\*\*](#)  
[\*\*FPDContentMarkItemSetParam\*\*](#)  
[\*\*FPDContentMarkAddMark\*\*](#)  
[\*\*FPDContentMarkCopy\*\*](#)  
[\*\*FPDContentMarkCountItems\*\*](#)  
[\*\*FPDContentMarkDeleteLastMark\*\*](#)  
[\*\*FPDContentMarkDeleteMark\*\*](#)  
[\*\*FPDContentMarkDestroy\*\*](#)  
[\*\*FPDContentMarkGetItem\*\*](#)  
[\*\*FPDContentMarkGetMCID\*\*](#)  
[\*\*FPDContentMarkHasMark\*\*](#)  
[\*\*FPDContentMarkIsNull\*\*](#)  
[\*\*FPDContentMarkLookupMark\*\*](#)

## Functions

### Functions summary

#### [\*\*FPDContentMarkAddMark\*\*](#)

Adds a content mark item.

#### [\*\*FPDContentMarkCopy\*\*](#)

Copies the source content mark to the specified one.

#### [\*\*FPDContentMarkCountItems\*\*](#)

Counts the number of content mark data in this object.

#### [\*\*FPDContentMarkDeleteLastMark\*\*](#)

Deletes the last content mark item.

#### [\*\*FPDContentMarkDeleteMark\*\*](#)

Deletes a content mark item.

#### [\*\*FPDContentMarkDestroy\*\*](#)

Destroys the PDF content mark object.

#### [\*\*FPDContentMarkGetItem\*\*](#)

Gets the content mark item by index.

#### [\*\*FPDContentMarkGetMCID\*\*](#)

Gets the marked-content identifier.

#### [\*\*FPDContentMarkHasMark\*\*](#)

Checks whether the content mark has a specified content mark item.

#### [\*\*FPDContentMarkIsNull\*\*](#)

Tests whether the content mark object is [NULL](#) or not.

#### [\*\*FPDContentMarkLookupMark\*\*](#)

Lookups a content mark item.

#### [\*\*FPDContentMarkNew\*\*](#)

Creates a new empty PDF content mark object.

### Functions detail

#### FPDContentMarkAddMark

##### Syntax

```
void FPDContentMarkAddMark (  
    FPD_ContentMark mark,
```

```
FS_LPCSTR tag,  
FPD_Object dict,  
FS_BOOL bDictNeedClone  
);
```

**Description**

Adds a content mark item.

**Parameter**

---

mark	[In] The input PDF content mark object.
tag	[In] The input name(tag) of the content mark item.
dict	[In] The parameter(attributes) dictionary of the content mark item.
bDictNeedClone	[In] Whether the input dictionary must be copied or not.

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 2099

**FPDContentMarkCopy****Syntax**

```
void FPDContentMarkCopy (  
    FPD_ContentMark mark,  
    FPD_ContentMark pSrcContentMark  
) ;
```

**Description**

Copies the source content mark to the specified one.

**Parameter**

---

mark	[In] The input PDF content mark object.
pSrcContentMark	[In] The input PDF content mark object to be copied.

---

**Return**

void.

**Head file reference**

fpd\_pageobjTempl.h: 2139

**Since**[SDK\\_LATEEST\\_VERSION > 8.3.1](#)**FPDContentMarkCountItems****Syntax**

```
FS_INT32 FPDContentMarkCountItems (
    FPD\_ContentMark mark
);
```

**Description**

Counts the number of content mark data in this object.

**Parameter**

---

mark	[In] The input PDF content mark object.
------	---

---

**Return**

The number of content mark data in this object.

**Head file reference**

fpd\_pageobjTempl.h: 2080

**FPDContentMarkDeleteLastMark****Syntax**

```
void FPDContentMarkDeleteLastMark (
    FPD\_ContentMark mark
);
```

**Description**

Deletes the last content mark item.

**Parameter**

---

mark	[In] The input PDF content mark object.
------	---

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 2121

## FPDContentMarkDeleteMark

### Syntax

```
void FPDContentMarkDeleteMark (
    FPD\_ContentMark mark,
    FS\_LPCSTR tag
);
```

### Description

Deletes a content mark item.

### Parameter

---

mark	[In] The input PDF content mark object.
------	---

---

tag	[In] The name(tag) of the content mark item.
-----	--

---

### Return

void

### Head file reference

fpd\_pageobjTempl.h: 2111

## FPDContentMarkDestroy

### Syntax

```
void FPDContentMarkDestroy (
    FPD\_ContentMark mark
);
```

### Description

Destroys the PDF content mark object.

### Parameter

---

mark	[In] The input PDF content mark object.
------	---

---

### Return

void

### Head file reference

fpd\_pageobjTempl.h: 2041

## FPDContentMarkGetItem

### Syntax

```
FPD_ContentMarkItem FPDContentMarkGetItem (
    FPD\_ContentMark mark,
    FS\_INT32 index
);
```

**Description**

Gets the content mark item by index.

**Parameter**

---

mark	[In] The input PDF content mark object.
index	[In] The zero-based content mark item index in the content mark data.

---

**Return**

A content mark item.

**Head file reference**

fpd\_pageobjTempl.h: 2089

**FPDContentMarkGetMCID****Syntax**

```
FS_INT32 FPDContentMarkGetMCID (
    FPD\_ContentMark mark
);
```

**Description**

Gets the marked-content identifier.

**Parameter**

---

mark	[In] The input PDF content mark object.
------	---

---

**Return**

The marked-content identifier.

**Head file reference**

fpd\_pageobjTempl.h: 2050

**FPDContentMarkHasMark****Syntax**

```
FS_BOOL FPDContentMarkHasMark (
    FPD\_ContentMark mark,
    FS\_LPCSTR tag
```

);

**Description**

Checks whether the content mark has a specified content mark item.

**Parameter**

---

mark	[In] The input PDF content mark object.
------	---

---

tag	[In] The name(tag) of the content mark item.
-----	--

---

**Return**

Non-zero means it has, otherwise it has not.

**Head file reference**

fpd\_pageobjTempl.h: 2059

**FPDContentMarkIsNull****Syntax**

```
FS_BOOL FPDContentMarkIsNull (
    FPD\_ContentMark mark
);
```

**Description**

Tests whether the content mark object is [NULL](#) or not.

**Parameter**

---

mark	[In] The input PDF content mark object.
------	---

---

**Return**

Non-zero means [NULL](#) , otherwise not [NULL](#) .

**Head file reference**

fpd\_pageobjTempl.h: 2130

**FPDContentMarkLookupMark****Syntax**

```
FS_BOOL FPDContentMarkLookupMark (
    FPD\_ContentMark mark,
    FS\_LPCSTR tag,
    FPD\_Object* outDict
);
```

**Description**

Lookups a content mark item.

**Parameter**

mark	[In] The input PDF content mark object.
tag	[In] The name(tag) of the content mark item.
outDict	[Out] It receives the parameter(attributes) dictionary.

**Return**

Non-zero means found one, otherwise found none.

**Head file reference**

fpd\_pageobjTempl.h: 2069

**FPDContentMarkNew****Syntax**

FPD\_ContentMark FPDContentMarkNew (void );

**Description**

Creates a new empty PDF content mark object.

**Return**

A new empty PDF content mark object.

**Head file reference**

fpd\_pageobjTempl.h: 2032

**FPD\_ContentMarkItem****Return from Used by****Description**

The content mark item for tagged pdf. See [FPDContentMarkItemNew](#) , [FPDContentMarkItemDestroy](#) .

**Returned from**

[FPDContentMarkGetItem](#)  
[FPDContentMarkItemNew](#)

## Used by

[FPDContentMarkItemDestroy](#)  
[FPDContentMarkItemGetName](#)  
[FPDContentMarkItemGetParam](#)  
[FPDContentMarkItemGetParamType](#)  
[FPDContentMarkItemSetName](#)  
[FPDContentMarkItemSetParam](#)

## Enumerations

### Enumerations summary

#### [FPD\\_MarkItemParamType](#)

Parameter type enumeration of content mark item. See  
[FPDContentMarkItemGetParamType](#) , [FPDContentMarkItemSetParam](#) .

### Enumerations detail

#### FPD\_MarkItemParamType

##### Syntax

```
enum FPD_MarkItemParamType{  
    NoneItem,  
    PropertiesDict,  
    DirectDict,  
    MCID  
};
```

##### Description

Parameter type enumeration of content mark item. See  
[FPDContentMarkItemGetParamType](#) , [FPDContentMarkItemSetParam](#) .

##### Head file reference

fpd\_pageobjExpT.h: 147

##### **NoneItem**

None parameters.

##### **PropertiesDict**

The dictionary defined by named resource in the Properties sub-dictionary of the current resource dictionary.

##### **DirectDict**

The dictionary may be written inline in the content stream as a direct object.

##### **MCID**

The dictionary contains an MCID entry, which is an integer marked-content identifier that uniquely identifies the marked-content sequence within its content stream.

## Functions

## Functions summary

### [FPDContentMarkItemDestroy](#)

Destroys the PDF content mark item object.

### [FPDContentMarkItemGetName](#)

Gets the name(tag) of the content mark item.

### [FPDContentMarkItemGetParam](#)

Gets the parameter of the content mark item.

### [FPDContentMarkItemGetType](#)

Gets the parameter type of the content mark item.

### [FPDContentMarkItemNew](#)

Creates a new empty PDF content mark item object.

### [FPDContentMarkItemSetName](#)

Sets the name(tag) of the content mark item.

### [FPDContentMarkItemSetParam](#)

Sets the parameter of the content mark item.

## Functions detail

### FPDContentMarkItemDestroy

#### **Syntax**

```
void FPDContentMarkItemDestroy (
    FPD\_ContentMarkItem item
);
```

#### **Description**

Destroys the PDF content mark item object.

#### **Parameter**

---

item	[In] The input PDF content mark item object.
------	--

---

#### **Return**

void

#### **Head file reference**

fpd\_pageobjTempl.h: 1967

### FPDContentMarkItemGetName

#### **Syntax**

```
char* FPDContentMarkItemGetName (
    FPD\_ContentMarkItem item
);
```

#### **Description**

Gets the name(tag) of the content mark item.

**Parameter**

---

item	[In] The input PDF content mark item object.
------	--

---

**Return**

The name(tag) of the content mark item.

**Head file reference**

fpd\_pageobjTempl.h: 1976

**FPDContentMarkItemGetParam****Syntax**

```
void* FPDContentMarkItemGetParam (
    FPD\_ContentMarkItem item
);
```

**Description**

Gets the parameter of the content mark item.

**Parameter**

---

item	[In] The input PDF content mark item object.
------	--

---

**Return**

The parameter of the content mark item.

**Head file reference**

fpd\_pageobjTempl.h: 1985

**FPDContentMarkItemGetParamType****Syntax**

```
FPD_MarkItemParamType FPDContentMarkItemGetParamType (
    FPD\_ContentMarkItem item
);
```

**Description**

Gets the parameter type of the content mark item.

**Parameter**

---

item	[In] The input PDF content mark item object.
------	--

---

**Return**

The parameter type of the content mark item.

**Head file reference**

fpd\_pageobjTempl.h: 1985

**FPDContentMarkItemNew****Syntax**

```
FPD_ContentMarkItem FPDContentMarkItemNew (void );
```

**Description**

Creates a new empty PDF content mark item object.

**Return**

A new empty PDF content mark item object.

**Head file reference**

fpd\_pageobjTempl.h: 1958

**FPDContentMarkItemSetName****Syntax**

```
void FPDContentMarkItemSetName (
    FPD\_ContentMarkItem item,
    FS\_LPCSTR csName
);
```

**Description**

Sets the name(tag) of the content mark item.

**Parameter**

---

item	[In] The input PDF content mark item object.
------	--

---

csName	[In] The input new name(tag).
--------	-------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 2003

**FPDContentMarkItemSetParam****Syntax**

```
void FPDContentMarkItemSetParam (
```

---

```
FPD_ContentMarkItem item,
FPD_MarkItemParamType type,
void* param
);
```

**Description**

Sets the parameter of the content mark item.

**Parameter**

item	[In] The input PDF content mark item object.
type	[In] The input new parameter type.
param	[In] The input new parameter.

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 2013

## FPD\_Creator

[Return from Used by](#)

**Description**

PDF Creator: creating PDF file from a document object. See [FPDCreatorNew](#) , [FPDCreatorDestroy](#) .

**Returned from**

[FRDocGetPDFCreator](#)  
[FPDCreatorNew](#)  
[FPDCreatorSetOption](#)

**Used by**

[FPDCreatorCompress](#)  
[FPDCreatorCreate](#)  
[FPDCreatorCreate2](#)  
[FPDCreatorCreate3](#)  
[FPDCreatorDestroy](#)  
[FPDCreatorGetObjectOffset](#)  
[FPDCreatorGetObjectSize](#)  
[FPDCreatorReleaseCustomSecurityData](#)  
[FPDCreatorReleaseDRMSecurityData](#)

[FPDCreatorRemoveSecurity](#)  
[FPDCreatorSetCustomSecurity](#)  
[FPDCreatorSetDRMProgressiveEncryptHandler](#)  
[FPDCreatorSetDRMSecurity](#)  
[FPDCreatorSetOption](#)  
[FPDCreatorSetProgressiveEncryptHandler](#)

## Definitions

### Definitions summary

#### [FPDFCREATE\\_INCREMENTAL](#)

Incremental.

#### [FPDFCREATE\\_NO\\_ORIGINAL](#)

No original.

#### [FPDFCREATE\\_OBJECTSTREAM](#)

Use object stream creator.

#### [FPDFCREATE\\_PROGRESSIVE](#)

Use progressive creator.

### Definitions detail

#### FPDFCREATE\_INCREMENTAL

##### Syntax

```
#define FPDFCREATE_INCREMENTAL 1
```

##### Description

Incremental.

##### Group

[FPDCreatingFlag](#)

##### Head file reference

fpd\_serialExpT.h: 82

#### FPDFCREATE\_NO\_ORIGINAL

##### Syntax

```
#define FPDFCREATE_NO_ORIGINAL 2
```

##### Description

No original.

##### Group

[FPDCreatingFlag](#)

##### Head file reference

fpd\_serialExpT.h: 84

## FPDFCREATE\_OBJECTSTREAM

### Syntax

```
#define FPDFCREATE_OBJECTSTREAM 8
```

### Description

Use object stream creator.

### Group

[FPDCreatingFlag](#)

### Head file reference

fpd\_serialExpT.h: 88

## FPDFCREATE\_PROGRESSIVE

### Syntax

```
#define FPDFCREATE_PROGRESSIVE 4
```

### Description

Use progressive creator.

### Group

[FPDCreatingFlag](#)

### Head file reference

fpd\_serialExpT.h: 86

## Functions

### Functions summary

#### [FPDCreatorCompress](#)

Sets data compression. By default, FPD\_Creator use Flate compression for all data streams in release mode, but not in debug mode.

#### [FPDCreatorCreate](#)

Write the whole document into a new file (using Unicode file name). Unicode version.

#### [FPDCreatorCreate2](#)

Write the whole document into a new file (using local file name). Local version.

#### [FPDCreatorCreate3](#)

Write the whole document to a custom file access.

#### [FPDCreatorDestroy](#)

Destroys the PDF file Creator.

#### [FPDCreatorGetObjectOffset](#)

Gets object offset.

#### [FPDCreatorGetObjectSize](#)

Gets object size.

#### [FPDCreatorModifyR5Security](#)

Modifies security permissions for Revision 5 handler (AES 256)

#### **FPDCreatorNew**

creates PDF file from a document object

#### **FPDCreatorReleaseCustomSecurityData**

Release the custom security data generated by FPDCreatorSetCustomSecurity.

#### **FPDCreatorReleaseDRMProgressiveEncryptHandler**

Releases the progressive encrypt handler.

#### **FPDCreatorReleaseDRMSecurityData**

Release the custom security data generated by FPDCreatorSetDRMSecurity.

#### **FPDCreatorReleaseOption**

Releases the creator option.

#### **FPDCreatorReleaseProgressiveEncryptHandler**

Releases the progressive encrypt handler.

#### **FPDCreatorRemoveSecurity**

Removes security settings. The output file will not be encrypted.

#### **FPDCreatorSetCustomSecurity**

Sets security using custom security handler and custom encryption.

#### **FPDCreatorSetDRMProgressiveEncryptHandler**

Sets the progressive encrypt handler so that the creator can encrypt the content progressively.

#### **FPDCreatorSetDRMSecurity**

Sets security using custom security handler and custom encryption.

#### **FPDCreatorSetOption**

Sets the creator option so that the creator can decode or encode the content progressively.

#### **FPDCreatorSetProgressiveEncryptHandler**

Sets the progressive encrypt handler so that the creator can encrypt the content progressively.

#### **FPDCreatorSetStandardSecurity**

Sets security settings using standard security handler only. Can't be used with incremental update.

## Functions detail

### FPDCreatorCompress

#### Syntax

```
void FPDCreatorCompress (
    FPD_Creator pFPDCreator,
    FS_BOOL bEnable
);
```

#### Description

Sets data compression. By default, FPD\_Creator use Flate compression for all data streams in release mode, but not in debug mode.

#### Parameter

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

bEnable	[In] Whether to do data compressing.
---------	--------------------------------------

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 478

**FPDCreatorCreate****Syntax**

```
FS_BOOL FPDCreatorCreate (
    FPD\_Creator pFPDCreator,
    FS\_LPCWSTR filename,
    FS\_DWORD flags
);
```

**Description**

Write the whole document into a new file (using Unicode file name). Unicode version.

**Parameter**

---

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

---

filename	[In] The output filename.
----------	---------------------------

---

flags	[In] The creating flags.
-------	--------------------------

---

**Return**

Non-zero means success, otherwise failed.

**Head file reference**

fpd\_serialTempl.h: 559

**FPDCreatorCreate2****Syntax**

```
FS_BOOL FPDCreatorCreate2 (
    FPD\_Creator pFPDCreator,
    FS\_LPCSTR filename,
    FS\_DWORD flags
);
```

**Description**

Write the whole document into a new file (using local file name). Local version.

**Parameter**

---

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

---

filename	[In] The output filename.
----------	---------------------------

---

flags	[In] The creating flags.
-------	--------------------------

---

**Return**

Non-zero means success, otherwise failed.

**Head file reference**

fpd\_serialTempl.h: 570

**FPDCreatorCreate3****Syntax**

```
FS_BOOL FPDCreatorCreate3 (
    FPD\_Creator pFPDCreator,
    FS\_StreamWriteHandler* pFile,
    FS\_DWORD flags
);
```

**Description**

Write the whole document to a custom file access.

**Parameter**


---

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

---

pFile	[Out] The output file access.
-------	-------------------------------

---

flags	[In] The creating flags.
-------	--------------------------

---

**Return**

Non-zero means success, otherwise failed.

**Head file reference**

fpd\_serialTempl.h: 581

**FPDCreatorDestroy****Syntax**

```
void FPDCreatorDestroy (
    FPD\_Creator pFPDCreator
);
```

**Description**

Destroys the PDF file Creator.

**Parameter**

---

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

---

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 469

**FPDCreatorGetObjectOffset****Syntax**

```
FS_DWORD FPDCreatorGetObjectOffset (
    FPD\_Creator pFPDCreator,
    FS\_DWORD objnum
);
```

**Description**

Gets object offset.

**Parameter**

---

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

---

---

objnum	[In] The object number.
--------	-------------------------

---

**Return**

the object offset.

**Head file reference**

fpd\_serialTempl.h: 592

**Note:** If the object with specific number doesn't exist, the returned value will be zero. If incremental update is used, offsets and sizes are only valid for modified indirect objects.

**FPDCreatorGetObjectSize****Syntax**

```
FS_DWORD FPDCreatorGetObjectSize (
    FPD\_Creator pFPDCreator,
    FS\_DWORD objnum
);
```

**Description**

Gets object size.

**Parameter**


---

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

---

objnum	[In] The object number.
--------	-------------------------

---

**Return**

the object size.

**Head file reference**

fpd\_serialTempl.h: 603

**Note:** If the object with specific number doesn't exist, the returned value will be zero. If incremental update is used, offsets and sizes are only valid for modified indirect objects.

**FPDCreatorModifyR5Security****Syntax**

```
void FPDCreatorModifyR5Security (
    pFPDCreator,
    bPermissions,
    permissions,
    bEncryptMetadata,
    FS_BOOL bUserPassword,
    FS_LPCBYTE user_pass,
    FS_DWORD user_size,
    FS_BOOL bOwnerPassword,
    FS_LPCBYTE owner_pass,
    FS_DWORD owner_size
);
```

**Description**

Modifies security permissions for Revision 5 handler (AES 256)

**Parameter**


---

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

---

bPermissions	[In] Whether permission and EncryptMetadata changed.
--------------	--

---

permissions	[In] New permissions value.
-------------	-----------------------------

---

---

bEncryptMetadata	[In] New EncryptMetadata value.
bUserPassword	[In] Whether user password changed (owner password required).
user_pass	[In] The user password pointer.
user_size	[In] The length of the user password.
bOwnerPassword	[In] Whether owner password changed.
owner_pass	[In] The owner password pointer.
owner_size	[In] The length of the owner password.

---

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 507

**FPDCreatorNew****Syntax**

```
FPD_Creator FPDCreatorNew (
    FPD Document pDoc
);
```

**Description**

creates PDF file from a document object

**Parameter**


---

pDoc	[In] The PDF Document Object.
------	-------------------------------

---

**Return**

The PDF Creator.

**Head file reference**

fpd\_serialTempl.h: 460

**FPDCreatorReleaseCustomSecurityData****Syntax**

```
void FPDCreatorReleaseCustomSecurityData (
    FPD\_Creator pFPDCreator,
    void* cusSecurityData
);
```

**Description**

Release the custom security data generated by FPDCreatorSetCustomSecurity.

**Parameter**

---

pFPDCreator	[In] The input PDF file Creator.
cusSecurityData	[In] The custom security data to be released.

---

**Return**

The data need to be released.

**Head file reference**

fpd\_serialTempl.h: 531

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDCreatorReleaseDRMProgressiveEncryptHandler****Syntax**

```
void FPDCreatorReleaseDRMProgressiveEncryptHandler (
    FPD\_ProgressiveEncryptHandler handler
);
```

**Description**

Releases the progressive encrypt handler.

**Parameter**

---

handler	[In] The input progressive encrypt handler to be released.
---------	--

---

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 695

**Since**

[SDK\\_LATEEST\\_VERSION > 9.5](#)

## FPDCreatorReleaseDRMSecurityData

### Syntax

```
void FPDCreatorReleaseDRMSecurityData (
    FPD\_Creator pFPDCreator,
    void* cusSecurityData
);
```

### Description

Release the custom security data generated by FPDCreatorSetDRMSecurity.

### Parameter

---

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

---

cusSecurityData	[In] The custom security data to be released.
-----------------	---

---

### Return

The data need to be released.

### Head file reference

fpd\_serialTempl.h: 664

### Since

[SDK\\_LATEEST\\_VERSION > 9.5](#)

## FPDCreatorReleaseOption

### Syntax

```
void FPDCreatorReleaseOption (
    handler
);
```

### Description

Releases the creator option.

### Parameter

---

handler	[In] The input creator option.
---------	--------------------------------

---

### Return

void.

### Head file reference

fpd\_serialTempl.h: 643

### Since

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)**FPDCreatorReleaseProgressiveEncryptHandler****Syntax**

```
void FPDCreatorReleaseProgressiveEncryptHandler (
    FPD\_ProgressiveEncryptHandler handler
);
```

**Description**

Releases the progressive encrypt handler.

**Parameter**

---

handler	[In] The input progressive encrypt handler to be released.
---------	--

---

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 621

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.3](#)

**FPDCreatorRemoveSecurity****Syntax**

```
void FPDCreatorRemoveSecurity (
    FPD\_Creator pFPDCreator
);
```

**Description**

Removes security settings. The output file will not be encrypted.

**Parameter**

---

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

---

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 550

**Note:** Can't be used with incremental update.

## FPDCreatorSetCustomSecurity

### Syntax

```
void* FPDCreatorSetCustomSecurity (
    FPD\_Creator pFPDCreator,
    FPD\_Object pEncryptDict,
    FPD\_CryptoHandler pCryptoHandler,
    FS\_BOOL bEncryptMetadata
);
```

### Description

Sets security using custom security handler and custom encryption.

### Parameter

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

pEncryptDict	[In] The Encrypt dictionary.
--------------	------------------------------

pCryptoHandler	[In] The crypto handler.
----------------	--------------------------

bEncryptMetadata	[In] Whether to encrypt the metadata.
------------------	---------------------------------------

### Return

The data need to be released. You can invoke FPDCreatorReleaseCustomSecurityData.

### Head file reference

[fpd\\_serialTempl.h](#): 525

**Note:** Application should provide a full encryption dictionary (application can destroy it after this call), and a custom encryption handler.

### Since

[SDK LATEEST VERSION > 1.0](#)

## FPDCreatorSetDRMProgressiveEncryptHandler

### Syntax

```
FPD_ProgressiveEncryptHandler FPDCreatorSetDRMProgressiveEncryptHandler (
    FPD\_Creator pFPDCreator,
    FPD\_ProgressiveEncryptCallbacks callbacks
);
```

### Description

---

Sets the progressive encrypt handler so that the creator can encrypt the content progressively.

**Parameter**


---

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

---

callbacks	[In] The callback set for progressive encrypt handler.
-----------	--

---

**Return**

The progressive encrypt handler.

**Head file reference**

fpd\_serialTempl.h: 683

**Related method**

[FPDCreatorDRMReleaseProgressiveEncryptHandler](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 9.5](#)

**FPDCreatorSetDRMSecurity****Syntax**

```
void* FPDCreatorSetDRMSecurity (
    FPD\_Creator pFPDCreator,
    FPD\_Object pEncryptDict,
    FPD\_CryptoHandler pCryptoHandler,
    FS\_BOOL bEncryptMetadata
);
```

**Description**

Sets security using custom security handler and custom encryption.

**Parameter**


---

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

---

pEncryptDict	[In] The Encrypt dictionary.
--------------	------------------------------

---

pCryptoHandler	[In] The crypto handler.
----------------	--------------------------

---

bEncryptMetadata	[In] Whether to encrypt the metadata.
------------------	---------------------------------------

---

**Return**

The data need to be released. You can invoke FPDCreatorReleaseDRMSecurityData.

**Head file reference**

fpd\_serialTempl.h: 658

**Note:** Application should provide a full encryption dictionary (application can destroy it after this call), and a custom encryption handler.

**Since**[SDK LATEEST VERSION > 9.5](#)**FPDCreatorSetOption****Syntax**

```
FPD_CreatorOption FPDCreatorSetOption (
    FPD\_Creator pFPDCreator,
    FPD\_CreatorOptionCallbacks callbacks
);
```

**Description**

Sets the creator option so that the creator can decode or encode the content progressively.

**Parameter**

---

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

---

callbacks	[In] The callback set for creator option.
-----------	---

---

**Return**

The creator option.

**Head file reference**

fpd\_serialTempl.h: 636

**Related method**[FPDCreatorReleaseOption](#)**Since**[SDK LATEEST VERSION > 2.1.0.4](#)**FPDCreatorSetProgressiveEncryptHandler****Syntax**

```
FPD_ProgressiveEncryptHandler FPDCreatorSetProgressiveEncryptHandler (
    FPD\_Creator pFPDCreator,
    FPD\_ProgressiveEncryptCallbacks callbacks
);
```



**Description**

Sets the progressive encrypt handler so that the creator can encrypt the content progressively.

**Parameter**

---

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

---

callbacks	[In] The callback set for progressive encrypt handler.
-----------	--

---

**Return**

The progressive encrypt handler.

**Head file reference**

fpd\_serialTempl.h: 614

**Related method**

[FPDCreatorReleaseProgressiveEncryptHandler](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.3](#)

## FPDCreatorSetStandardSecurity

**Syntax**

```
void FPDCreatorSetStandardSecurity (
    pFPDCreator,
    permissions,
    user_pass,
    user_pass_len,
    FS_LPCBYTE owner_pass,
    FS_INT32 owner_pass_len,
    FS_INT32 cipher,
    FS_INT32 key_bytes,
    FS_BOOL bEncryptMetadata
);
```

**Description**

Sets security settings using standard security handler only. Can't be used with incremental update.

**Parameter**

---

pFPDCreator	[In] The input PDF file Creator.
-------------	----------------------------------

---

permissions	[In] The user permissions.
-------------	----------------------------

---

---

user_pass	[In] The user password.
user_pass_len	[In] The length of user password.
owner_pass	[In] The owner password.
owner_pass_len	[In] The length of owner password.
cipher	[In] The cipher type, RC4 or AES.
key_bytes	[In] The length of the document key.
bEncryptMetadata	[In] Whether to encrypt the metadata.

---

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 489

## FPD\_CreatorOption

[Return from Used by](#)

### Description

The [FPD\\_CREATOROPTION](#) object is used to decode or encode the stream progressively. It is returned by [FPDCreatorSetOption](#) and can be released by [FPDCreatorReleaseOption](#).

### Returned from

[FPDCreatorSetOption](#)

### Used by

[FPDCreatorSetOption](#)

### Callbacks

#### Callbacks summary

[FPDCreatorOptionGetTempFile](#)

A callback for creator option.

[FPDCreatorOptionReleaseTempFile](#)

A callback for creator option.

## Callbacks detail

### FPDCreatorOptionGetTempFile

#### Syntax

```
typedef FS_FileStream (*FPDCreatorOptionGetTempFile)(
    FS_LPVVOID clientData,
    FPD_Object obj
);
```

#### Description

A callback for creator option. It is called when the PDF creator is decoding or encoding content progressively. If the stream is very large, the plug-in can provide a temporary local file to store the data, or the creating may fail because of lacking of system memory. Foxit Reader will use memory if the plug-in returns null.

#### Parameter

clientData	[In] The user-supplied data.
------------	------------------------------

obj	[In] The object to be processed.
-----	----------------------------------

#### Return

The [FS FileStream](#) object used to store and read the temporary data.

#### Head file reference

fpd\_serialExpT.h: 246

#### Group

[FPD\\_CreatorOptionCallbacksRec](#)

#### Related method

[FPDCreatorSetOption](#)

#### Since

[SDK LATEEST VERSION > 2.1.0.4](#)

### FPDCreatorOptionReleaseTempFile

#### Syntax

```
typedef void (*FPDCreatorOptionReleaseTempFile)(
    FS_LPVVOID clientData,
    FS_FileStream fileStream
);
```

#### Description

A callback for creator option. It is called when the PDF creator completes decoding or encoding content progressively. Then the plug-in can release the [FS\\_FileStream](#) object.

#### Parameter

clientData	[In] The user-supplied data.
fileStream	[In] The <a href="#">FS_FileStream</a> object to be released.

#### Return

void.

#### Head file reference

fpd\_serialExpT.h: 262

#### Group

[FPD\\_CreatorOptionCallbacksRec](#)

#### Related method

[FPDCreatorSetOption](#)

#### Since

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

## FPD\_CryptoHandler

### Used by

#### Description

Abstract class for PDF cryptographic operations (encryption and decryption). This class works with security handler which provides algorithm and key info.

#### Used by

[FPDCreatorSetCustomSecurity](#)  
[FPDCreatorSetDRMSecurity](#)

## FPD\_DefaultAppearance

### Return from Used by

#### Description

Default appearance interpreter, serves for DA entry in form field dictionary.

#### Returned from

### [FPDDefaultAppearanceNew](#)

Used by

[FPDFFormControlGetDefaultAppearance](#)  
[FPDIInterFormGetDefaultAppearance](#)  
[FPDDefaultAppearanceDestroy](#)  
[FPDDefaultAppearanceGetColor](#)  
[FPDDefaultAppearanceGetColorInclueCMYK](#)  
[FPDDefaultAppearanceGetColorString](#)  
[FPDDefaultAppearanceGetFont](#)  
[FPDDefaultAppearanceGetFontString](#)  
[FPDDefaultAppearanceGetTextMatrix](#)  
[FPDDefaultAppearanceGetTextMatrixString](#)  
[FPDDefaultAppearanceHasColor](#)  
[FPDDefaultAppearanceHasFont](#)  
[FPDDefaultAppearanceHasTextMatrix](#)  
[FPDDefaultAppearanceSetColor](#)  
[FPDDefaultAppearanceSetFont](#)  
[FPDDefaultAppearanceSetTextMatrix](#)

## Functions

Functions summary

### [FPDDefaultAppearanceDestroy](#)

Destroys default appearance interpreter.

### [FPDDefaultAppearanceGetColor](#)

Gets the color (not including CMYK).

### [FPDDefaultAppearanceGetColorInclueCMYK](#)

Gets the color (including CMYK).

### [FPDDefaultAppearanceGetColorString](#)

Gets the color instruction in the appearance string.

### [FPDDefaultAppearanceGetFont](#)

Gets the font.

### [FPDDefaultAppearanceGetFontString](#)

Gets the font instruction in the appearance string.

### [FPDDefaultAppearanceGetTextMatrix](#)

Gets the text matrix.

### [FPDDefaultAppearanceGetTextMatrixString](#)

Gets the text matrix instruction in the appearance string.

### [FPDDefaultAppearanceHasColor](#)

Checks whether a color has been used in the appearance.

### [FPDDefaultAppearanceHasFont](#)

Checks whether a font has been used in the appearance.

### [FPDDefaultAppearanceHasTextMatrix](#)

Checks whether a text matrix has been used in the appearance.

### [FPDDefaultAppearanceNew](#)

Creates from a default appearance string that containing a sequence of valid page-content graphics or text state operators that define such properties as the field's text size and color.

### [FPDDefaultAppearanceSetColor](#)

Sets the color.

**FPDDefaultAppearanceSetFont**

Sets the font name tag and size of the font.

**FPDDefaultAppearanceSetTextMatrix**

Sets the text matrix.

## Functions detail

### FPDDefaultAppearanceDestroy

**Syntax**

```
void FPDDefaultAppearanceDestroy (
    FPD_DefaultAppearance defAppearance
);
```

**Description**

Destroys default appearance interpreter.

**Parameter**

---

defAppearance	[In] The input default appearance interpreter.
---------------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3678

### FPDDefaultAppearanceGetColor

**Syntax**

```
void FPDDefaultAppearanceGetColor (
    FPD_DefaultAppearance defAppearance,
    FS_ARGB* pClr,
    FS_INT32* pIclrType,
    FS_BOOL bstrokingOperation
);
```

**Description**

Gets the color (not including CMYK).

**Parameter**

---

defAppearance	[In] The input default appearance interpreter.
---------------	--

---

pClr	[Out] It receives the color.
------	------------------------------

---

pIclrType	[Out] It receives the color type.
-----------	-----------------------------------

---

---

bStrokingOperation	[In] Whether the stroking color to be fetched.
--------------------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3738

**FPDDefaultAppearanceGetColorInclueCMYK****Syntax**

```
void FPDDefaultAppearanceGetColorInclueCMYK (
    FPD\_DefaultAppearance defAppearance,
    FS\_INT32* pIclrType,
    FS\_FLOAT fc[4],
    FS\_BOOL bStrokingOperation
);
```

**Description**

Gets the color (including CMYK).

**Parameter**


---

defAppearance	[In] The input default appearance interpreter.
---------------	--

---

pIclrType	[Out] It receives the color type.
-----------	-----------------------------------

---

fc[4]	[Out] It receives the color values of components.
-------	---

---

bStrokingOperation	[In] Whether the stroking color to be fetched.
--------------------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3749

**FPDDefaultAppearanceGetString****Syntax**

```
void FPDDefaultAppearanceGetString (
    FPD\_DefaultAppearance defAppearance,
    FS\_BOOL bStrokingOperation,
    FS\_BytString* outColorString
);
```

**Description**

Gets the color instruction in the appearance string.

**Parameter**

---

defAppearance	[In] The input default appearance interpreter.
---------------	--

---

bstrokingOperation	[In] Whether the stroking color instruction to be fetched.
--------------------	--

---

outColorString	[Out] The color instruction in the appearance string.
----------------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3738

**FPDDefaultAppearanceGetFont****Syntax**

```
void FPDDefaultAppearanceGetFont (
    FPD\_DefaultAppearance defAppearance,
    FS\_ByteString* outFontNameTag,
    FS\_FLOAT* outFontSize
);
```

**Description**

Gets the font.

**Parameter**

---

defAppearance	[In] The input default appearance interpreter.
---------------	--

---

outFontNameTag	[Out] It receives font name tag of font.
----------------	--

---

outFontSize	[Out] It receives the font size of the font.
-------------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3696

**FPDDefaultAppearanceGetFontString**

**Syntax**

```
void FPDDefaultAppearanceGetFontString (
    FPD\_DefaultAppearance defAppearance,
    FS\_ByteString* outFontString
);
```

**Description**

Gets the font instruction in the appearance string.

**Parameter**

---

defAppearance	[In] The input default appearance interpreter.
---------------	--

---

outFontString	[Out] It receives the font instruction in the appearance string.
---------------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3696

**FPDDefaultAppearanceGetTextMatrix****Syntax**

```
FS_AffineMatrix FPDDefaultAppearanceGetTextMatrix (
    FPD\_DefaultAppearance defAppearance
);
```

**Description**

Gets the text matrix.

**Parameter**

---

defAppearance	[In] The input default appearance interpreter.
---------------	--

---

**Return**

The text matrix.

**Head file reference**

fpd\_docTempl.h: 3794

**FPDDefaultAppearanceGetTextMatrixString****Syntax**

```
void FPDDefaultAppearanceGetTextMatrixString (
    FPD\_DefaultAppearance defAppearance,
    FS\_ByteString* outMatrixString
```

);

**Description**

Gets the text matrix instruction in the appearance string.

**Parameter**

---

defAppearance	[In] The input default appearance interpreter.
---------------	--

---

outMatrixString	[Out] It receives the text matrix instruction in the appearance string.
-----------------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3794

**FPDDefaultAppearanceHasColor****Syntax**

```
FS_BOOL FPDDefaultAppearanceHasColor (
    FPD_DefaultAppearance defAppearance,
    FS_BOOL bstrokingOperation
);
```

**Description**

Checks whether a color has been used in the appearance.

**Parameter**

---

defAppearance	[In] The input default appearance interpreter.
---------------	--

---

bstrokingOperation	[In] Whether the stroking color to be checked.
--------------------	--

---

**Return**

Non-zero means having one, otherwise having none.

**Head file reference**

fpd\_docTempl.h: 3728

**FPDDefaultAppearanceHasFont****Syntax**

```
FS_BOOL FPDDefaultAppearanceHasFont (
```

```
FPD_DefaultAppearance defAppearance  
);
```

**Description**

Checks whether a font has been used in the appearance.

**Parameter**

---

defAppearance	[In] The input default appearance interpreter.
---------------	--

---

**Return**

[TRUE](#) for a font having been used in the appearance, otherwise not.

**Head file reference**

fpd\_docTempl.h: 3687

**FPDDefaultAppearanceHasTextMatrix****Syntax**

```
FS_BOOL FPDDefaultAppearanceHasTextMatrix (  
    FPD_DefaultAppearance defAppearance  
);
```

**Description**

Checks whether a text matrix has been used in the appearance.

**Parameter**

---

defAppearance	[In] The input default appearance interpreter.
---------------	--

---

**Return**

[TRUE](#) for a text matrix having been used in the appearance, otherwise not.

**Head file reference**

fpd\_docTempl.h: 3785

**FPDDefaultAppearanceNew****Syntax**

```
FPD_DefaultAppearance FPDDefaultAppearanceNew (  
    FS_LPCSTR szAppearance  
);
```

**Description**

Creates from a default appearance string that containing a sequence of valid page-content graphics or text state operators that define such properties as the field's text size and color.

**Parameter**

---

szAppearance	[In] The input default appearance string.
--------------	---

---

**Return**

Default appearance interpreter, serves for DA entry in form field dictionary.

**Head file reference**

fpd\_docTempl.h: 3667

**FPDDefaultAppearanceSetColor****Syntax**

```
void FPDDefaultAppearanceSetColor (
    FPD\_DefaultAppearance defAppearance,
    FS\_ARGB clr,
    FS\_INT32 iClrType,
    FS\_BOOL bStrokingOperation
);
```

**Description**

Sets the color.

**Parameter**

---

defAppearance	[In] The input default appearance interpreter.
---------------	--

---

---

clr	[In] The input color.
-----	-----------------------

---

---

iClrType	[In] The input color type.
----------	----------------------------

---

---

bStrokingOperation	[In] Whether the stroking color to be set.
--------------------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3773

**FPDDefaultAppearanceSetFont****Syntax**

```
void FPDDefaultAppearanceSetFont (
    FPD\_DefaultAppearance defAppearance,
    FS\_LPSTR szFontNameTag,
```

```
FS_FLOAT fFontSize  
);
```

**Description**

Sets the font name tag and size of the font.

**Parameter**

defAppearance	[In] The input default appearance interpreter.
szFontNameTag	[In] The font name tag of the font.
fFontSize	[In] The size of the font.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3717

**FPDDefaultAppearanceSetTextMatrix****Syntax**

```
void FPDDefaultAppearanceSetTextMatrix (  
    FPD_DefaultAppearance defAppearance,  
    FS_AffineMatrix matrix  
) ;
```

**Description**

Sets the text matrix.

**Parameter**

defAppearance	[In] The input default appearance interpreter.
matrix	[In] The input text matrix.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3813

## FPD\_Dest

[Return from Used by](#)

### Description

Corresponding to a PDF Dest array. It contains a reference to a page, a ctangle on that page, and information specifying how to adjust the view to fit the window's size and shape. See [FPDDestNew](#) , [FPDDestDestroy](#) , [FPDBookmarkGetDest](#) , [FPDACTIONGetDest](#) , [FPDACTIONSetDest](#) , [FPDLINKGetDest](#) .

### Returned from

[FPDDestNew](#)

### Used by

[FPDACTIONGetDest](#)  
[FPDBookmarkGetDest](#)  
[FPDLINKGetDest](#)  
[FPDDestDestroy](#)  
[FPDDestGetPageIndex](#)  
[FPDDestGetPageObjNum](#)  
[FPDDestGetParam](#)  
[FPDDestGetPDFObject](#)  
[FPDDestGetRemoteName](#)  
[FPDDestGetZoomMode](#)

### Definitions

#### Definitions summary

##### [FPD\\_ZOOM\\_FITHORZ](#)

Fit horizontal.

##### [FPD\\_ZOOM\\_FITPAGE](#)

Fit page.

##### [FPD\\_ZOOM\\_FITRECT](#)

Fit rectangle.

##### [FPD\\_ZOOM\\_FITVERT](#)

Fit vertical.

##### [FPD\\_ZOOM\\_XYZ](#)

Left, top, zoom.

#### Definitions detail

##### [FPD\\_ZOOM\\_FITHORZ](#)

###### Syntax

```
#define FPD_ZOOM_FITHORZ 3
```

###### Description

Fit horizontal. With the vertical coordinate top positioned at the top edge of the window and the contents of the page magnified just enough to fit the entire width of the page within the window.

**Group**

[FPDDestinationZoomMode](#)

**Head file reference**

fpd\_docExpT.h: 348

## FPD\_ZOOM\_FITPAGE

**Syntax**

#define FPD\_ZOOM\_FITPAGE 2

**Description**

Fit page. With its contents magnified just enough to fit the entire page within the window both horizontally and vertically.

**Group**

[FPDDestinationZoomMode](#)

**Head file reference**

fpd\_docExpT.h: 340

## FPD\_ZOOM\_FITRECT

**Syntax**

#define FPD\_ZOOM\_FITRECT 5

**Description**

Fit rectangle. With its contents magnified just enough to fit the rectangle specified by the coordinates left, bottom, right, and top entirely within the window both horizontally and vertically.

**Group**

[FPDDestinationZoomMode](#)

**Head file reference**

fpd\_docExpT.h: 364

## FPD\_ZOOM\_FITVERT

**Syntax**

#define FPD\_ZOOM\_FITVERT 4

**Description**

Fit vertical. With the horizontal coordinate left positioned at the left edge of the window and the contents of the page magnified just enough to fit the tire height of the page within the window.

**Group**

[FPDDestinationZoomMode](#)

**Head file reference**

fpd\_docExpT.h: 356

## FPD\_ZOOM\_XYZ

**Syntax**

#define FPD\_ZOOM\_XYZ 1

**Description**

Left, top, zoom. With the coordinates (left, top) positioned at the upper-left corner of the window and the contents of the page magnified by the factor zoom.

**Group**

[FPDDestinationZoomMode](#)

**Head file reference**

fpd\_docExpT.h: 333

## Functions

### Functions summary

[\*\*FPDDestDestroy\*\*](#)

Destroys a PDF destination object.

[\*\*FPDDestGetPageIndex\*\*](#)

Gets zero-based index of the page in the document.

[\*\*FPDDestGetPageObjNum\*\*](#)

Gets the object number of the page.

[\*\*FPDDestGetParam\*\*](#)

Gets a param.

[\*\*FPDDestGetPDFObject\*\*](#)

Gets the PDF object of the destination.

[\*\*FPDDestGetRemoteName\*\*](#)

Gets the remote name of named destination.

[\*\*FPDDestGetZoomMode\*\*](#)

Gets the zoom mode of the destination.

[\*\*FPDDestNew\*\*](#)

Creates a PDF destination object from a PDF object.

### Functions detail

#### FPDDestDestroy

**Syntax**

```
void FPDDestDestroy (
    FPD\_Dest dest
);
```

**Description**

Destroys a PDF destination object.

**Parameter**

---

dest	[In] The input PDF destination object.
------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 962

**FPDDestGetPageIndex****Syntax**

```
FS_INT32 FPDDestGetPageIndex (
    FPD\_Document doc,
    FPD\_Dest dest
);
```

**Description**

Gets zero-based index of the page in the document.

**Parameter**

---

doc	[In] The PDF document.
-----	------------------------

---

---

dest	[In] A PDF destination object.
------	--------------------------------

---

**Return**

The zero-based index of the page referred to.

**Head file reference**

fpd\_docTempl.h: 981

**FPDDestGetPageObjNum****Syntax**

```
FS_DWORD FPDDestGetPageObjNum (
    FPD\_Dest dest
);
```

**Description**

Gets the object number of the page.

**Parameter**

---

dest	[In] A PDF destination object.
------	--------------------------------

---

**Return**

The object number of the page.

**Head file reference**

fpd\_docTempl.h: 991

**FPDDestGetParam****Syntax**

```
double FPDDestGetParam (
    FPD_Dest dest,
    FS_INT32 index
);
```

**Description**

Gets a param.

**Parameter**

---

dest	[In] A PDF destination object.
------	--------------------------------

---

index	[In] The zero-based index of the param.
-------	---

---

**Return**

The param by index.

**Head file reference**

fpd\_docTempl.h: 1009

**FPDDestGetPDFObject****Syntax**

```
void FPDDestGetPDFObject (
    FPD_Dest dest,
    FPD_Object* pObject
);
```

**Description**

Gets the PDF object of the destination.

**Parameter**

---

dest	[In] A PDF destination object.
------	--------------------------------

---

pObject	[Out] It receives the PDF object of the destination.
---------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1019

**FPDDestGetRemoteName****Syntax**

```
void FPDDestGetRemoteName (
    FPD\_Dest dest,
    FS\_ByteString* pStrResult
);
```

**Description**

Gets the remote name of named destination.

**Parameter**

---

dest	[In] A PDF destination object.
------	--------------------------------

---

pStrResult	[Out] It receives the remote name of named destination.
------------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 971

**FPDDestGetZoomMode****Syntax**

```
FS_INT32 FPDDestGetZoomMode (
    FPD\_Dest dest
);
```

**Description**

Gets the zoom mode of the destination.

#### Parameter

---

dest	[In] A PDF destination object.
------	--------------------------------

---

#### Return

The zoom mode of the destination.

#### Head file reference

fpd\_docTempl.h: 1000

## FPDDestNew

#### Syntax

```
FPD_Dest FPDDestNew (
    FPD Object obj
);
```

#### Description

Creates a PDF destination object from a PDF object.

#### Parameter

---

obj	[In] The input PDF object.
-----	----------------------------

---

#### Return

A PDF destination object

#### Head file reference

fpd\_docTempl.h: 953

## FPD\_Dictionary

### Description

A FPD\_Dictionary is an associative table whose elements are pairs of objects:

- The first element of a pair is the key, which is always a name object, a sequence of characters ginning with the forward slash (/) character.
- The second element is the Cos object representing the value.

See Section 3.2.6 in the PDF Reference for details.

### Functions

## Functions summary

### **FPDictionaryAddValue**

Adds a key-value pair to the dictionary, assuming there is no duplicated key existing. This is a function for quickly building up the whole dictionary, but should be used with care. If duplicate key happens, only the first value will prevail.

### **FPDictionaryGetArray**

Gets an array object specified by key.

### **FPDictionaryGetBoolean**

Gets the boolean data for the element specified by key with a default boolean value.

### **FPDictionaryGetCount**

Gets the number of elements in the dictionary.

### **FPDictionaryGetDict**

Gets a dictionary object specified by key.

### **FPDictionaryGetElement**

Gets direct reference to the object (include reference), Don't free the returned object.

### **FPDictionaryGetElementValue**

Gets a direct or referred indirect object, Don't free the returned object.

### **FPDictionaryGetFloat**

Gets a floating-point value for the element specified by key.

### **FPDictionaryGetInteger**

Gets the integer data for the element specified by key.

### **FPDictionaryGetInteger2**

Gets the integer data for the element specified by key with a default integer value.

### **FPDictionaryGetMatrix**

Gets a matrix for the element specified by key.

### **FPDictionaryGetNextElement**

Gets a direct reference to the element and move the position to next. Don't free the returned object.

### **FPDictionaryGetNumber**

Gets the number data for the element specified by key.

### **FPDictionaryGetRect**

Gets a rectangle for the element specified by key.

### **FPDictionaryGetPosition**

Gets the position for the first element.

### **FPDictionaryGetStream**

Gets a stream object specified by key.

### **FPDictionaryGetString**

Gets the string data for the element specified by key.

### **FPDictionaryGetUnicodeText**

Gets the Unicode string data for the element specified by key with a character mapping.

### **FPDictionaryIdentical**

Compares value with another object.

### **FPDictionaryKeyExist**

Tests whether the element specified by key is exist.

### **FPDictionaryNew**

Creates an empty dictionary.

### **FPDictionaryRemoveAt**

Removes the element specified by key.

### **FPDictionaryReplaceKey**

Replaces the key of the element specified by key with new key string.

**FPDictionarySetAt**

Sets element data. Please note all elements will be managed with the dictionary object, so the object pointer must NOT be freed by caller.

**FPDictionarySetAtBoolean**

Sets a boolean value of boolean object for the element specified by key.

**FPDictionarySetAtInteger**

Sets an integer of number object for the element specified by key.

**FPDictionarySetAtMatrix**

Sets a matrix for the element specified by key.

**FPDictionarySetAtName**

Sets a string of name object for the element specified by key.

**FPDictionarySetAtNumber**

Sets a FIX24.8 of number object for the element specified by key.

**FPDictionarySetAtRect**

Sets a rectangle for the element specified by key.

**FPDictionarySetAtReference2ToDoc**

Sets a reference object for the element specified by key.

**FPDictionarySetAtReference2ToFDFDoc**

Sets a reference object for the element specified by key.

**FPDictionarySetAtReferenceToDoc**

Sets a reference object for the element specified by key.

**FPDictionarySetAtReferenceToFDFDoc**

Sets a reference object for the element specified by key.

**FPDictionarySetAsString**

Sets a string of string object for the element specified by key.

## Functions detail

### FPDictionaryAddValue

#### Syntax

```
void FPDictionaryAddValue (
    FPD_Object objDict,
    const char* key,
    FPD_Object obj
);
```

#### Description

Adds a key-value pair to the dictionary, assuming there is no duplicated key existing. This is a function for quickly building up the whole dictionary, but should be used with care. If duplicate key happens, only the first value will prevail.

#### Parameter

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key.
-----	---------------------

---

obj	[In] The input value.
-----	-----------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1171

**FPDDictionaryGetArray****Syntax**

```
FPD_Object FPDDictionaryGetArray (
    FPD_Object objDict,
    const char* key
);
```

**Description**

Gets an array object specified by key.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

**Return**

An array object.

**Head file reference**

fpd\_objsTempl.h: 920

**FPDDictionaryGetBoolean****Syntax**

```
FS_BOOL FPDDictionaryGetBoolean (
    FPD_Object objDict,
    const char* key
);
```

**Description**

Gets the boolean data for the element specified by key with a default boolean value.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

---

key	[In] The input key string.
-----	----------------------------

---

**Return**

A boolean value for the specified element.

**Head file reference**

fpd\_objsTempl.h: 880

**FPDDictionaryGetCount****Syntax**

```
FS_INT32 FPDDictionaryGetCount (
    FPD Object objDict
);
```

**Description**

Gets the number of elements in the dictionary.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

**Return**

The number of elements in the dictionary.

**Head file reference**

fpd\_objsTempl.h: 1162

**FPDDictionaryGetDict****Syntax**

```
FPD_Object FPDDictionaryGetDict (
    FPD Object objDict,
    const char* key
);
```

**Description**

Gets a dictionary object specified by key.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

---

key	[In] The input key string.
-----	----------------------------

---

**Return**

A dictionary object.

**Head file reference**

fpd\_objsTempl.h: 900

**FPDDictionaryGetElement****Syntax**

```
FPD_Object FPDDictionaryGetElement (
    FPD_Object objDict,
    const char* key
);
```

**Description**

Gets direct reference to the object (include reference), Don't free the returned object.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

**Return**

A pointer to the object (include reference).

**Head file reference**

fpd\_objsTempl.h: 815

**FPDDictionaryGetValue****Syntax**

```
FPD_Object FPDDictionaryGetValue (
    FPD_Object objDict,
    const char* key
);
```

**Description**

Gets a direct or referred indirect object, Don't free the returned object.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

**Return**

A pointer to direct or referred indirect object.

**Head file reference**

fpd\_objsTempl.h: 826

**FPDDictionaryGetFloat****Syntax**

```
FS_FLOAT FPDDictionaryGetFloat (
    FPD Object objDict,
    const char* key
);
```

**Description**

Gets a floating-point value for the element specified by key.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

**Return**

A floating-point value.

**Head file reference**

fpd\_objsTempl.h: 950

**FPDDictionaryGetInteger****Syntax**

```
FS_INT32 FPDDictionaryGetInteger (
    FPD Object objDict,
    const char* key
);
```

**Description**

Gets the integer data for the element specified by key.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

**Return**

An integer value for the specified element.

**Head file reference**

fpd\_objsTempl.h: 859

**FPDDictionaryGetInteger2****Syntax**

```
FS_INT32 FPDDictionaryGetInteger2 (
    FPD Object objDict,
    const char* key,
    FS INT32 defaultInt
);
```

**Description**

Gets the integer data for the element specified by key with a default integer value.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

defaultInt	[In] The default integer value.
------------	---------------------------------

---

**Return**

An integer value for the specified element.

**Head file reference**

fpd\_objsTempl.h: 869

**FPDDictionaryGetMatrix****Syntax**

```
FS_AffineMatrix FPDDictionaryGetMatrix (
    FPD Object objDict,
    const char* key
);
```

**Description**

Gets a matrix for the element specified by key.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

**Return**

A matrix.

**Head file reference**

fpd\_objsTempl.h: 940

**FPDDictionaryGetNextElement****Syntax**

```
FPD_Object FPDDictionaryGetNextElement (
    FPD\_Object objDict,
    FS\_POSITION* outPos,
    FS\_BytString* outKey
);
```

**Description**

Gets a direct reference to the element and move the position to next. Don't free the returned object.

**Parameter**


---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

outPos	[In/Out] Input current position and receive the next position.
--------	--

---

outKey	[Out] It receives the current key string.
--------	---

---

**Return**

The direct reference to the current element.

**Head file reference**

fpd\_objsTempl.h: 979

**FPDDictionaryGetNumber****Syntax**

```
FS_FLOAT FPDDictionaryGetNumber (
    FPD\_Object objDict,
    const char* key
);
```

**Description**

Gets the number data for the element specified by key.

**Parameter**

objDict	[In] The input PDF dictionary object.
key	[In] The input key string.

**Return**

A FIX24.8 value for the specified element.

**Head file reference**

fpd\_objsTempl.h: 890

**FPDDictionaryGetRect****Syntax**

```
FS_FloatRect FPDDictionaryGetRect (
    FPD Object objDict,
    const char* key
);
```

**Description**

Gets a rectangle for the element specified by key.

**Parameter**

objDict	[In] The input PDF dictionary object.
key	[In] The input key string.

**Return**

A rectangle.

**Head file reference**

fpd\_objsTempl.h: 930

**FPDDictionaryGetStartPosition****Syntax**

```
FS_POSITION FPDDictionaryGetStartPosition (
    FPD Object objDict
);
```

**Description**

Gets the position for the first element.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

**Return**

The position for the first element.

**Head file reference**

fpd\_objsTempl.h: 970

**FPDDictionaryGetStream****Syntax**

```
FPD_Object FPDDictionaryGetStream (
    FPD_Object objDict,
    const char* key
);
```

**Description**

Gets a stream object specified by key.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

**Return**

A stream object.

**Head file reference**

fpd\_objsTempl.h: 910

**FPDDictionaryGetString****Syntax**

```
void FPDDictionaryGetString (
    FPD_Object objDict,
    const char* key,
    FS_ByteString* outString
);
```

**Description**

Gets the string data for the element specified by key.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

outString	[Out] A byte string for the specified element.
-----------	--

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 837

**FPDDictionaryGetUnicodeText****Syntax**

```
void FPDDictionaryGetUnicodeText (
    FPD Object objDict,
    const char* key,
    FS WideString* outUnicodeText
);
```

**Description**

Gets the Unicode string data for the element specified by key with a character mapping.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

outUnicodeText	[Out] An Unicode string for the specified element.
----------------	--

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 848

**FPDDictionaryIdentical****Syntax**

```
FS_BOOL FPDDictionaryIdentical (
    FPD Object objDict,
    FPD Object otherDict
);
```

**Description**

Ccompares value with another object.

**Parameter**

objDict	[In] The input PDF dictionary object.
otherDict	[In] The another dictionary.

**Return**

Non-zero means identical, otherwise not identical.

**Head file reference**

fpd\_objsTempl.h: 1152

**FPDDictionaryKeyExist****Syntax**

```
FS_BOOL FPDDictionaryKeyExist (
    FPD Object objDict,
    const char* key
);
```

**Description**

Tests whether the element specified by key is exist.

**Parameter**

objDict	[In] The input PDF dictionary object.
key	[In] The input key string.

**Return**

Non-zero means exist, otherwise not.

**Head file reference**

fpd\_objsTempl.h: 960

**FPDDictionaryNew**

**Syntax**

```
FPD_Object FPDDictionaryNew (void );
```

**Description**

Creates an empty dictionary.

**Return**

A PDF dictionary object.

**Head file reference**

fpd\_objsTempl.h: 805

**Related method**

[FPDOObjectDestroy](#)

**FPDDictionaryRemoveAt****Syntax**

```
void FPDDictionaryRemoveAt (
    FPD Object objDict,
    const char* key
);
```

**Description**

Removes the element specified by key.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1131

**FPDDictionaryReplaceKey****Syntax**

```
void FPDDictionaryReplaceKey (
    FPD Object objDict,
    const char* oldkey,
    const char* newKey
);
```

**Description**

Replaces the key of the element specified by key with new key string.

**Parameter**

objDict	[In] The input PDF dictionary object.
oldkey	[In] The old key string.
newKey	[In] The new key string.

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1141

**FPDDictionarySetAt****Syntax**

```
void FPDDictionarySetAt (
    FPD\_Object objDict,
    const char* key,
    FPD\_Object obj
);
```

**Description**

Sets element data. Please note all elements will be managed with the dictionary object, so the object pointer must NOT be freed by caller.

**Parameter**

objDict	[In] The input PDF dictionary object.
key	[In] The input key string.
obj	[In] The input element data. param[in] objs The indirect object collection, required if obj is an indirect object. In this case, a reference object will be created and inserted into the dictionary. If not required, sets it as NULL.

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 991

**Since**

[SDK LATEEST VERSION > 1.0](#)

## FPDDictionarySetAtBoolean

**Syntax**

```
void FPDDictionarySetAtBoolean (
    FPD Object objDict,
    const char* key,
    FS\_BOOL value
);
```

**Description**

Sets a boolean value of boolean object for the element specified by key.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

value	[In] The input boolean value.
-------	-------------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1120

## FPDDictionarySetAtInteger

**Syntax**

```
void FPDDictionarySetAtInteger (
    FPD Object objDict,
    const char* key,
    FS\_INT32 i
);
```

**Description**

Sets an integer of number object for the element specified by key.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
i	[In] The input integer.

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1028

**FPDDictionarySetAtMatrix****Syntax**

```
void FPDDictionarySetAtMatrix (
    FPD_Object objDict,
    const char* key,
    FS_AffineMatrix matrix
);
```

**Description**

Sets a matrix for the element specified by key.

**Parameter**

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

key	[In] The input key string.
-----	----------------------------

matrix	[In] The input matrix.
--------	------------------------

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1109

**FPDDictionarySetAtName****Syntax**

```
void FPDDictionarySetAtName (
    FPD_Object objDict,
    const char* key,
    FS_LPCSTR szName
);
```

**Description**

Sets a string of name object for the element specified by key.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

szName	[In] The name string.
--------	-----------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1006

**FPDDictionarySetAtNumber****Syntax**

```
void FPDDictionarySetAtNumber (
    FPD Object objDict,
    const char* key,
    FS FLOAT f
);
```

**Description**

Sets a FIX24.8 of number object for the element specified by key.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

f	[In] The input FIX24.8 value.
---	-------------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1039

**FPDDictionarySetAtRect**

**Syntax**

```
void FPDDictionarySetAtRect (  
    FPD_Object objDict,  
    const char* key,  
    FS_FloatRect rect  
) ;
```

**Description**

Sets a rectangle for the element specified by key.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

rect	[In] The input rectangle.
------	---------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1098

**FPDDictionarySetAtReference2ToDoc****Syntax**

```
void FPDDictionarySetAtReference2ToDoc (  
    FPD_Object objDict,  
    const char* key,  
    FPD_Document doc,  
    FPD_Object obj  
) ;
```

**Description**

Sets a reference object for the element specified by key.

**Parameter**

---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

doc	[In] The indirect objects collection for the reference object.
-----	--

---

---

obj	[In] The referred object pointer for the reference object.
-----	--

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1074

**FPDDictionarySetAtReference2ToFDFDoc****Syntax**

```
void FPDDictionarySetAtReference2ToFDFDoc (
    FPD Object objDict,
    const char* key,
    FDF Document doc,
    FPD Object obj
);
```

**Description**

Sets a reference object for the element specified by key.

**Parameter**


---

objDict	[In] The input PDF dictionary object.
---------	---------------------------------------

---

key	[In] The input key string.
-----	----------------------------

---

doc	[In] The indirect objects collection for the reference object.
-----	--

---

obj	[In] The referred object pointer for the reference object.
-----	--

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1086

**FPDDictionarySetAtReferenceToDoc****Syntax**

```
void FPDDictionarySetAtReferenceToDoc (
    FPD Object objDict,
    const char* key,
    FDF Document doc,
    FS DWORD objNum
);
```

**Description**

Sets a reference object for the element specified by key.

**Parameter**

objDict	[In] The input PDF dictionary object.
key	[In] The input key string.
doc	[In] The indirect objects collection for the reference object.
objNum	[In] The referred object number for the reference object.

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1050

**FPDDictionarySetAtReferenceToFDFDoc****Syntax**

```
void FPDDictionarySetAtReferenceToFDFDoc (  
    FPD Object objDict,  
    const char* key,  
    FDF Document doc,  
    FS DWORD objNum  
) ;
```

**Description**

Sets a reference object for the element specified by key.

**Parameter**

objDict	[In] The input PDF dictionary object.
key	[In] The input key string.
doc	[In] The indirect objects collection for the reference object.
objNum	[In] The referred object number for the reference object.



**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1062

**FPDDictionarySetAtString****Syntax**

```
void FPDDictionarySetAtString (
    FPD Object objDict,
    const char* key,
    FS ByteString str
);
```

**Description**

Sets a string of string object for the element specified by key.

**Parameter**

objDict	[In] The input PDF dictionary object.
key	[In] The input key string.
str	[In] The input string.

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1017

## FPD\_DocJSActions

[Return from Used by](#)

### Description

A JavaScript action causes a script to be complied and executed by the JavaScript interpreter. See [FPDDocJSActionsNew](#) , [FPDDocJSActionsDestroy](#) .

### Returned from

[FPDDocJSActionsNew](#)

### Used by



[\*\*FPDDocJSActionsCountJSActions\*\*](#)  
[\*\*FPDDocJSActionsDestroy\*\*](#)  
[\*\*FPDDocJSActionsFindJSAction\*\*](#)  
[\*\*FPDDocJSActionsGetDocument\*\*](#)  
[\*\*FPDDocJSActionsGetJSAction\*\*](#)  
[\*\*FPDDocJSActionsGetJSAction2\*\*](#)  
[\*\*FPDDocJSActionsRemoveJSAction\*\*](#)  
[\*\*FPDDocJSActionsSetJSAction\*\*](#)

## Functions

### Functions summary

#### [\*\*FPDDocJSActionsCountJSActions\*\*](#)

Gets the count of JavaScript actions.

#### [\*\*FPDDocJSActionsDestroy\*\*](#)

Destroys document-level JavaScript actions.

#### [\*\*FPDDocJSActionsFindJSAction\*\*](#)

Finds a JavaScript action. return the position (zero based index), -1 means not found.

#### [\*\*FPDDocJSActionsGetDocument\*\*](#)

Gets the PDF document.

#### [\*\*FPDDocJSActionsGetJSAction\*\*](#)

Gets a JavaScript action with the position and the name.

#### [\*\*FPDDocJSActionsGetJSAction2\*\*](#)

Gets a JavaScript action with the name.

#### [\*\*FPDDocJSActionsNew\*\*](#)

Creates document-level JavaScript actions from a PDF document.

#### [\*\*FPDDocJSActionsRemoveJSAction\*\*](#)

Removes a JavaScript action.

#### [\*\*FPDDocJSActionsSetJSAction\*\*](#)

Sets a JavaScript action.

### Functions detail

#### FPDDocJSActionsCountJSActions

##### Syntax

```
FS_INT32 FPDDocJSActionsCountJSActions (  
    FPD\_DocJSActions jsActions  
)
```

##### Description

Gets the count of JavaScript actions.

##### Parameter

---

jsActions	[In] The input document-level JavaScript actions.
-----------	---

---

##### Return

The count of JavaScript actions.

**Head file reference**

fpd\_docTempl.h: 2375

**FPDDocJSActionsDestroy****Syntax**

```
void FPDDocJSActionsDestroy (
    FPD\_DocJSActions jsActions
);
```

**Description**

Destroys document-level JavaScript actions.

**Parameter**

---

jsActions	[In] The input document-level JavaScript actions.
-----------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2366

**FPDDocJSActionsFindJSAction****Syntax**

```
FS_INT32 FPDDocJSActionsFindJSAction (
    FPD\_DocJSActions jsActions,
    FS\_LPCWSTR wszName
);
```

**Description**

Finds a JavaScript action. return the position (zero based index), -1 means not found.

**Parameter**

---

jsActions	[In] The input document-level JavaScript actions.
-----------	---

---

wszName	[In] The name of the JavaScript action.
---------	---

---

**Return**

The zero-based index of the JavaScript action.

**Head file reference**

fpd\_docTempl.h: 2418

**FPDDocJSActionsGetDocument****Syntax**

```
FPD_Document FPDDocJSActionsGetDocument (
    FPD\_DocJSActions jsActions
);
```

**Description**

Gets the PDF document.

**Parameter**

---

jsActions	[In] The input document-level JavaScript actions.
-----------	---

---

**Return**

The PDF document.

**Head file reference**

fpd\_docTempl.h: 2439

**FPDDocJSActionsGetJSAction****Syntax**

```
void FPDDocJSActionsGetJSAction (
    FPD\_DocJSActions jsActions,
    FS\_INT32 index,
    FS\_LPCWSTR wszName,
    FPD\_Action* outAction
);
```

**Description**

Gets a JavaScript action with the position and the name.

**Parameter**

---

jsActions	[In] The input document-level JavaScript actions.
-----------	---

---

---

index	[In] The zero-based JavaScript action index.
-------	--

---

---

wszName	[In] The name of the JavaScript action.
---------	---

---

---

outAction	[Out] A JavaScript action.
-----------	----------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2384

**FPDDocJSActionsGetJSAction2****Syntax**

```
void FPDDocJSActionsGetJSAction2 (
    FPD\_DocJSActions jsActions,
    FS\_LPCWSTR wszName,
    FPD\_Action* outAction
);
```

**Description**

Gets a JavaScript action with the name.

**Parameter**

---

jsActions	[In] The input document-level JavaScript actions.
-----------	---

---

wszName	[In] The name of the JavaScript action.
---------	---

---

outAction	[Out] A JavaScript action.
-----------	----------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2396

**FPDDocJSActionsNew****Syntax**

```
FPD_DocJSActions FPDDocJSActionsNew (
    FPD\_Document doc
);
```

**Description**

Creates document-level JavaScript actions from a PDF document.

**Parameter**

---

doc	[In] The input PDF document.
-----	------------------------------

---

**Return**

Document-level JavaScript actions.

**Head file reference**

fpd\_docTempl.h: 2357

**FPDDocJSActionsRemoveJSAction****Syntax**

```
FS_BOOL FPDDocJSActionsRemoveJSAction (
    FPD\_DocJSActions jsActions,
    FS\_INT32 index
);
```

**Description**

Removes a JavaScript action.

**Parameter**

---

jsActions	[In] The input document-level JavaScript actions.
index	[In] The zero-based index of JavaScript action to be removed.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 2429

**FPDDocJSActionsSetJSAction****Syntax**

```
FS_INT32 FPDDocJSActionsSetJSAction (
    FPD\_DocJSActions jsActions,
    FS\_LPCWSTR wszName,
    FPD\_Action action
);
```

**Description**

Sets a JavaScript action.

**Parameter**

---

jsActions	[In] The input document-level JavaScript actions.
-----------	---

---

---

wszName	[In] The name of the JavaScript action.
action	[In] The JavaScript action.

---

**Return**

The zero-based index of the JavaScript action.

**Head file reference**

fpd\_docTempl.h: 2407

## FPD\_Document

### [Return from Used by](#)

### Description

The underlying PDF representation of a document. Each PDF document contains, among other things:

- - A tree of pages( [FPD\\_Page](#) ).
- - (Optional) Trees of bookmarks and articles.
- - (Optional) Information and security dictionaries.

The FPD\_Document object is the hidden object behind every FR\_Document. Through FPD\_Document objects, your application can perform most of the menu items for pages on Foxit Phantom (delete, replace, and so on). Thumbnails and Bookmarks can be created and deleted through this object. You can set and retrieve document information fields through this object as well. The first page in a FPD\_Document is page 0. See [FPDDocNew](#) , [FPDDocDestroy](#) , [FPDDocOpen](#) , [FPDDocOpenMemDocument](#) , [FPDDocClose](#) , [FPDPageGetDocument](#) .

### Returned from

[FRDocGetPDDoc](#)  
[FPDAnnotListGetDocument](#)  
[FPDDocJSActionsGetDocument](#)  
[FPDFFontGetPDFDoc](#)  
[FPDFFormGetDocument](#)  
[FPDImageGetDocument](#)  
[FPDInterFormGetDocument](#)  
[FPDOCContextGetDocument](#)  
[FPDOCPropertiesGetDocument](#)  
[FPDPageGetDocument](#)  
[FPDParseGetDocument](#)  
[FPDPatternGetPDDoc](#)  
[FPDShadingPatternGetPDDoc](#)  
[FPDTilingPatternGetPDDoc](#)  
[FPDDocNew](#)  
[FPDDocOpen](#)  
[FPDDocOpenMemDocument](#)

## Used by

[FRAppIsPDF2Doc](#)  
[FRAppRefreshLayerPanelView](#)  
[FRDocFromPDDoc](#)  
[FRDocOpenFromPDDoc](#)  
[FRDocOpenFromPDDoc2](#)  
[FPDFDFDocExportAnnotToPDFPage](#)  
[FPDAActionSetAction](#)  
[FPDActionGetDest](#)  
[FPDActionGetURI](#)  
[FPDActionInsertOCGStates](#)  
[FPDActionInsertRendition](#)  
[FPDActionInsertSubAction](#)  
[FPDActionNew2](#)  
[FPDActionNew3](#)  
[FPDActionReplaceOCGStates](#)  
[FPDActionSetAnnot](#)  
[FPDActionSetJavaScript](#)  
[FPDActionSetJavaScriptW](#)  
[FPDArrayAddReference2ToDoc](#)  
[FPDArrayAddReference>ToDoc](#)  
[FPDBookmarkGetDest](#)  
[FPDBookmarkGetFirstChild](#)  
[FPDBookmarkGetNextSibling](#)  
[FPDCColorSpaceLoad](#)  
[FPDConnectedInfoIsConnectedPDF2](#)  
[FPDConnectedInfoNew](#)  
[FPDCreatorNew](#)  
[FPDDestGetPageIndex](#)  
[FPDDictionarySetAtReference2ToDoc](#)  
[FPDDictionarySetAtReference>ToDoc](#)  
[FPDDocJSActionsNew](#)  
[FPDFFileSpecSetEmbeddedFile](#)  
[FPDFFontGetStockFont](#)  
[FPDFFontNew](#)  
[FPDFFormNew](#)  
[FPDImageNew](#)  
[FPDInterFormNew](#)  
[FPDLinkCountLinks](#)  
[FPDLinkGetDest](#)  
[FPDLinkGetLink](#)  
[FPDLinkGetLinkAtPoint](#)  
[FPDNameTreeLookupNamedDest](#)  
[FPDNameTreeNew2](#)  
[FPDNameTreeSetValue](#)  
[FPDOObjectCloneRefToDoc](#)  
[FPDOCContextNew](#)  
[FPDOCPropertiesNew](#)  
[FPDPageGetPageText](#)  
[FPDPageGetPageText\\_Unicode](#)  
[FPDPageLoad](#)  
[FPDParseParseIndirectObject](#)  
[FPDParseParseIndirectObjectAt](#)  
[FPDReferenceNew](#)  
[FPDReferenceSetRefToDoc](#)

[FPDRenderContextNew2](#)  
[FPDRenditionSetMediaClipFile](#)  
[FPDShadingPatternNew](#)  
[FPDTilingPatternNew](#)  
[FPDTilingPatternNewII](#)  
[FPDUnencryptedWrapperCreatorNew](#)  
[FPDWrapperCreatorNew](#)  
[FPDWrapperDocNew](#)  
[FPDDocAddFXFont](#)  
[FPDDocAddIndirectObject](#)  
[FPDDocAddStandardFont](#)  
[FPDDocAddWindowsFont](#)  
[FPDDocAddWindowsFontW](#)  
[FPDDocBuildResourceList](#)  
[FPDDocCheckSignature](#)  
[FPDDocClearModified](#)  
[FPDDocClose](#)  
[FPDDocContinueLoad](#)  
[FPDDocConvertIndirectObjects](#)  
[FPDDocCreateNewPage](#)  
[FPDDocDeleteIndirectObject](#)  
[FPDDocDeletePage](#)  
[FPDDocDestroy](#)  
[FPDDocEnumPages](#)  
[FPDDocGetID](#)  
[FPDDocGetIndirectObject](#)  
[FPDDocGetIndirectObjsCount](#)  
[FPDDocGetIndirectType](#)  
[FPDDocGetInfo](#)  
[FPDDocGetInfoObjNum](#)  
[FPDDocGetLastobjNum](#)  
[FPDDocGetNextAssoc](#)  
[FPDDocGetPage](#)  
[FPDDocGetPageContentModify](#)  
[FPDDocGetPageCount](#)  
[FPDDocGetPageIndex](#)  
[FPDDocGetParser](#)  
[FPDDocGetReviewType](#)  
[FPDDocGetRoot](#)  
[FPDDocGetRootObjNum](#)  
[FPDDocGetPosition](#)  
[FPDDoc GetUserPermissions](#)  
[FPDDocImportExternalObject](#)  
[FPDDocImportIndirectObject](#)  
[FPDDocInsertDocumentAtPos](#)  
[FPDDocInsertIndirectObject](#)  
[FPDDocIsFormStream](#)  
[FPDDocIsModified](#)  
[FPDDocIsOwner](#)  
[FPDDocIsPortfolio](#)  
[FPDDocLoadColorSpace](#)  
[FPDDocLoadDoc](#)  
[FPDDocLoadFont](#)  
[FPDDocLoadFontFile](#)  
[FPDDocLoadImageF](#)  
[FPDDocLoadPattern](#)  
[FPDDocQuickSearch](#)

[FPDDocReleaseIndirectObject](#)  
[FPDDocReloadFileStreams](#)  
[FPDDocReloadPages](#)  
[FPDDocRemoveUR3](#)  
[FPDDocSave](#)  
[FPDDocSave2](#)  
[FPDDocSetID](#)  
[FPDDocSetInfoObjNum](#)  
[FPDDocSetRootObjNum](#)  
[FPDDocStartProgressiveLoad](#)

## Definitions

### Definitions summary

#### [FPD PERM ANNOT FORM](#)

bit 6.

#### [FPD PERM ASSEMBLE](#)

bit 11.

#### [FPD PERM EXTRACT](#)

bit 5.

#### [FPD PERM EXTRACT ACCESS](#)

bit 10.

#### [FPD PERM FILL FORM](#)

bit 9.

#### [FPD PERM MODIFY](#)

bit 4.

#### [FPD PERM PRINT](#)

bit 3.

#### [FPD PERM PRINT HIGH](#)

bit 12.

#### [FPD SAVE DEFAULT](#)

Default.

#### [FPD SAVE INCREMENTAL](#)

Incremental.

#### [FPD SAVE NO ORIGINAL](#)

No original.

#### [FPD LOADERR ERROR](#)

Error of any kind, without specific reason.

#### [FPD LOADERR FILE](#)

File access error.

#### [FPD LOADERR FORMAT](#)

Not PDF format.

#### [FPD LOADERR MEMORY](#)

Out of memory.

#### [FPD LOADERR NOTFOUND](#)

Not found.

#### [FPD LOADERR PARAM](#)

Parameter error.

#### [FPD LOADERR PASSWORD](#)

Incorrect password.

**FPD\_LOADERR\_STATUS**

Not in correct status.

**FPD\_LOADERR\_SUCCESS**

Load document successfully.

**FPD\_LOADERR\_TOBECONTINUED**

To be continued.

**Definitions detail****FPD\_PERM\_ANNOT\_FORM****Syntax**

```
#define FPD_PERM_ANNOT_FORM 0x0020
```

**Description**

bit 6. Add or modify text annotations, fill in interactive form fields, and, bit 4 is also set, create or modify interactive form fields (including signature fields).

**Group**

[FPDDocPermissions](#)

**Head file reference**

fpd\_docExpT.h: 160

**FPD\_PERM\_ASSEMBLE****Syntax**

```
#define FPD_PERM_ASSEMBLE 0x0400
```

**Description**

bit 11. (Revision 3 or greater) Assemble the document (insert, rotate, or delete pages and create bookmarks or thumbnail images), even if bit 4 is clear.

**Group**

[FPDDocPermissions](#)

**Head file reference**

fpd\_docExpT.h: 181

**FPD\_PERM\_EXTRACT****Syntax**

```
#define FPD_PERM_EXTRACT 0x0010
```

**Description**

bit 5.

- - (Revision 2) Copy or otherwise extract text and graphics from the document, including extracting text and graphics (in support of accessibility to users with disabilities or for other purposes).
- - (Revision 3 or greater) Copy or otherwise extract text and graphics from the document by operations other than that controlled by bit 10.

**Group**

[FPDDocPermissions](#)

**Head file reference**

fpd\_docExpT.h: 153

## FPD\_PERM\_EXTRACT\_ACCESS

**Syntax**

```
#define FPD_PERM_EXTRACT_ACCESS 0x0200
```

**Description**

bit 10. (Revision 3 or greater) Extract text and graphics (in support of accessibility to users with disabilities or for other purposes).

**Group**

[FPDDocPermissions](#)

**Head file reference**

fpd\_docExpT.h: 174

## FPD\_PERM\_FILL\_FORM

**Syntax**

```
#define FPD_PERM_FILL_FORM 0x0100
```

**Description**

bit 9. (Revision 3 or greater) Fill in existing interactive form fields (including signature fields), even if bit 6 is clear.

**Group**

[FPDDocPermissions](#)

**Head file reference**

fpd\_docExpT.h: 167

## FPD\_PERM MODIFY

**Syntax**

```
#define FPD_PERM MODIFY 0x0008
```

**Description**

bit 4. Modify the contents of the document by operations other than those controlled by bits 6, 9, and 11.

**Group**

[FPDDocPermissions](#)

**Head file reference**

fpd\_docExpT.h: 141

**FPD\_PERM\_PRINT****Syntax**

#define FPD\_PERM\_PRINT 0x0004

**Description**

bit 3.

- - (Revision 2) Print the document.
- - (Revision 3 or greater) Print the document (possibly not at the highest quality level, depending on whether bit 12 is also set).

**Group**

[FPDDocPermissions](#)

**Head file reference**

fpd\_docExpT.h: 134

**FPD\_PERM\_PRINT\_HIGH****Syntax**

#define FPD\_PERM\_PRINT\_HIGH 0x0800

**Description**

bit 12. (Revision 3 or greater) Print the document to a representation from which a faithful digital copy of the PDF content could be generated. When this bit is clear (and bit 3 is set), printing is limited to a low-level presentation of the appearance, possibly of degraded quality. (See implementation note 25 in Appendix H.)

**Group**

[FPDDocPermissions](#)

**Head file reference**

fpd\_docExpT.h: 191

**FPD\_SAVE\_DEFAULT**

**Syntax**

```
#define FPD_SAVE_DEFAULT 0
```

**Description**

Default.

**Group**

[FPDDocSaveFlags](#)

**Head file reference**

fpd\_docExpT.h: 205

## FPD\_SAVE\_INCREMENTAL

**Syntax**

```
#define FPD_SAVE_INCREMENTAL 1
```

**Description**

Incremental.

**Group**

[FPDDocSaveFlags](#)

**Head file reference**

fpd\_docExpT.h: 207

## FPD\_SAVE\_NO\_ORIGINAL

**Syntax**

```
#define FPD_SAVE_NO_ORIGINAL 2
```

**Description**

No original.

**Group**

[FPDDocSaveFlags](#)

**Head file reference**

fpd\_docExpT.h: 209

## FPD\_LOADERR\_ERROR

**Syntax**

```
#define FPD_LOADERR_ERROR 2
```

**Description**

Error of any kind, without specific reason.

**Group**  
[FPDLoadErrCodes](#)

**Head file reference**  
fpd\_docExpT.h: 90

## FPD\_LOADERR\_FILE

**Syntax**  
`#define FPD_LOADERR_FILE 5`

**Description**  
File access error.

**Group**  
[FPDLoadErrCodes](#)

**Head file reference**  
fpd\_docExpT.h: 99

## FPD\_LOADERR\_FORMAT

**Syntax**  
`#define FPD_LOADERR_FORMAT 4`

**Description**  
Not PDF format.

**Group**  
[FPDLoadErrCodes](#)

**Head file reference**  
fpd\_docExpT.h: 96

## FPD\_LOADERR\_MEMORY

**Syntax**  
`#define FPD_LOADERR_MEMORY 1`

**Description**  
Out of memory.

**Group**  
[FPDLoadErrCodes](#)

**Head file reference**

fpd\_docExpT.h: 87

## FPD\_LOADERR\_NOTFOUND

### Syntax

#define FPD\_LOADERR\_NOTFOUND 9

### Description

Not found.

### Group

[FPDLoadErrCodes](#)

### Head file reference

fpd\_docExpT.h: 111

## FPD\_LOADERR\_PARAM

### Syntax

#define FPD\_LOADERR\_PARAM 6

### Description

Parameter error.

### Group

[FPDLoadErrCodes](#)

### Head file reference

fpd\_docExpT.h: 102

## FPD\_LOADERR\_PASSWORD

### Syntax

#define FPD\_LOADERR\_PASSWORD 3

### Description

Incorrect password.

### Group

[FPDLoadErrCodes](#)

### Head file reference

fpd\_docExpT.h: 93

## FPD\_LOADERR\_STATUS

**Syntax**

```
#define FPD_LOADERR_STATUS 7
```

**Description**

Not in correct status.

**Group**

[FPDLoadErrCodes](#)

**Head file reference**

fpd\_docExpT.h: 105

## FPD\_LOADERR\_SUCCESS

**Syntax**

```
#define FPD_LOADERR_SUCCESS 0
```

**Description**

Load document successfully.

**Group**

[FPDLoadErrCodes](#)

**Head file reference**

fpd\_docExpT.h: 84

## FPD\_LOADERR\_TOBECONTINUED

**Syntax**

```
#define FPD_LOADERR_TOBECONTINUED 8
```

**Description**

To be continued.

**Group**

[FPDLoadErrCodes](#)

**Head file reference**

fpd\_docExpT.h: 108

## Functions

### Functions summary

**[FPDDocAddFXFont](#)**

Adds the Foxit GE font to the PDF document. It will return a PDF font object.

**[FPDDocAddIndirectObject](#)**

Adds an object to indirect object list. The new object number is returned.

**FPDDocAddStandardFont**

Adds a Type1 base-14 font to the PDF document.

**FPDDocAddWindowsFont**

Imports a Windows font into the PDF document.

**FPDDocAddWindowsFontW**

Imports a Windows font into the PDF document.

**FPDDocBuildResourceList**

Build a list of all resources of specified type in this document. On return, the array contains pointers to FPD\_Object objects.

**FPDDocCheckSignature**

Checks whether the document is signed or not.

**FPDDocClearModified**

Clears the modified flag.

**FPDDocClose**

Closes the document returned by [FPDDocOpen](#) (), [FPDDocStartProgressiveLoad](#) () or [FPDDocOpenMemDocument](#) ().

**FPDDocContinueLoad**

Continue loading a PDF document. You must call [FPDDocClose](#) () once for every successful open.

**FPDDocConvertIndirectObjects**

Converts stream of dictionary members of an object to be indirect objects.

**FPDDocCreateNewPage**

Creates a new page in the PDF document. Return the created page.

**FPDDocDeleteEnumPageHandler**

Deletes the user-supplied page enumeration handler.

**FPDDocDeleteIndirectObject**

Deletes an indirect object. Use this function with caution!

**FPDDocDeletePage**

Deletes specified page in the PDF document.

**FPDDocDestroy**

Destroys the document created by [FPDDocNew](#) . When Foxit Reader opens the doc, it is taken over and don't destroy it.

**FPDDocEnumPages**

Enumerates pages with user-supplied page enumeration handler.

**FPDDocGenerateFileID**

Generates the ID of the PDF document.

**FPDDocGetID**

Gets ID strings of the document. Could be empty.

**FPDDocGetIndirectObject**

Loads an indirect object by an object number.

**FPDDocGetIndirectObjsCount**

Gets the indirect object count.

**FPDDocGetIndirectType**

Gets type of an document, without loading the object content.

**FPDDocGetInfo**

Gets document info dictionary from a PDF document. Could be NULL.

**FPDDocGetInfoObjNum**

Gets the number of info object.

**FPDDocGetLastobjNum**

Gets last assigned indirect object number.

**FPDDocGetNextAssoc**

Accesses the indirect object of current position, and move the position to next.

**FPDDocGetPage**

Gets page dictionary of a specified page of the document. Page index starts from zero for the first page.

**FPDDocGetPageContentModify**

Gets a modifiable content stream for a page in the PDF document.

**FPDDocGetPageCount**

Gets number of pages in this document.

**FPDDocGetPageIndex**

Gets the page index from a indirect object number in the document.

**FPDDocGetParser**

Gets the PDF file parser associated with a [FPD\\_Document](#) object.

**FPDDocGetReviewType**

Gets the review type. 0:normal, 1:share, 2:email.

**FPDDocGetRoot**

Gets root dictionary (document catalog dictionary) from a PDF document.

**FPDDocGetRootObjNum**

Gets the number of root object.

**FPDDocGetPosition**

Gets the start position of the indirect objects.

**FPDDoc GetUserPermissions**

Gets the user permission of the document.

**FPDDocImportExternalObject**

Imports an object from external object collection as a new document. If the external object refers to other external indirect objects, those indirect objects are also imported. The mapping from old object number to new object number is updated during the import process.

**FPDDocImportIndirectObject**

Imports an object from its binary format. This is used when the PDF is fetched in "Progressive Downloading" fashion. After this function call, the data buffer can be destroyed. This object must not be encrypted.

**FPDDocInsertDocumentAtPos**

Inserts the document into the original document at the specified position.

**FPDDocInsertIndirectObject**

Inserts an indirect object with specified object number. This is used when the PDF is fetched in "Progressive Downloading" fashion, or parsing FDF. If an indirect object with the same object number exists, the previous one will be destroyed.

**FPDDocIsFormStream**

Checks if an indirect object is a form stream or not, without actually loading the object.

**FPDDocIsModified**

Checks if any of the indirect objects are modified, since loading from parser, or last ClearModified.

**FPDDocIsOwner**

Whether the user has the owner permission of the document.

**FPDDocIsPortfolio**

Checks whether the document is a portfolio document or not.

**FPDDocLoadColorSpace**

Loads a color space from a PDF object in the document.

**FPDDocLoadDoc**

Internally used

**FPDDocLoadFont**

Loads a font from font dictionary in the document.

**FPDDocLoadFontFile**

Loads a PDF stream accessor from a PDF stream in the document.

**FPDDocLoadImageF**

Loads an image from a PDF object in the document.

**FPDDocLoadPattern**

Loads a pattern from a PDF object in the document.

**FPDDocNew**

Creates a new empty document.

**FPDDocNewEnumPageHandler**

Creates user-supplied page enumeration handler.

**FPDDocOpen**

Creates a new document from an existing PDF file. You must call [FPDDocClose](#) () once for every successful open.

**FPDDocOpenMemDocument**

Creates a document from memory block. You must call [FPDDocClose](#) () once for every successful open.

**FPDDocQuickSearch**

Quick search an pattern for specified page in the PDF document.

**FPDDocReleaseIndirectObject**

Sometimes, for saving memory space, we can release a loaded indirect object. However, use this with caution because the object pointer will become invalid.

**FPDDocReloadFileStreams**

Reload all file based stream when we do reparsing.

**FPDDocReloadPages**

Reload page list. This can be used when document is progressively downloaded.

**FPDDocRemoveUR3**

Removes the UR3 signature.

**FPDDocSave**

Writes the whole document into a new file.

**FPDDocSave2**

Writes the whole document into a new file.

**FPDDocSetID**

Sets the PDF file ID.

**FPDDocSetInfoObjNum**

Sets the info object number in the PDF document. Must be called for an empty document.

**FPDDocSetRootObjNum**

Sets the root object number in the PDF document. Must be called for an empty document.

**FPDDocStartProgressiveLoad**

Document loading is a progressive process. It might take a long time to load a document, especially when a file is corrupted, FPDFEMB will try to recover the document contents by scanning the whole file. If "pause" parameter is provided, this function may return FPDFERR\_TOBECONTINUED any time during the document loading.

When [FPD\\_LOADERR\\_TOBECONTINUED](#) is returned, the "outPDDoc" parameter will still receive a valid document handle, however, no further operations can be performed on the document, except the [FPDDocContinueLoad](#) () function call, which resume the document loading.

## Functions detail

### FPDDocAddFXFont

**Syntax**

```
FPD_Font FPDDocAddFXFont (
    FPD\_Document doc,
    FPD\_FXFont pFXFont,
    FS\_INT32 charset,
    FS\_BOOL bVert
);
```

**Description**

Adds the Foxit GE font to the PDF document. It will return a PDF font object.

**Parameter**

---

doc	[In] The input PDF document.
-----	------------------------------

---

pFXFont	[In] The input Foxit GE font object.
---------	--------------------------------------

---

charset	[In] The input Charset ID.
---------	----------------------------

---

bVert	[In] Whether it's a vertical writing font. For CJK charsets only.
-------	---

---

**Return**

The PDF font object.

**Head file reference**

fpd\_docTempl.h: 660

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.0.0](#)

**FPDDocAddIndirectObject****Syntax**

```
FS_DWORD FPDDocAddIndirectObject (
    FPD\_Document doc,
    FPD\_Object obj
);
```

**Description**

Adds an object to indirect object list. The new object number is returned.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

---

obj	[In] The input object.
-----	------------------------

---

**Return**

The new object number.

**Head file reference**

fpd\_docTempl.h: 458

**FPDDocAddStandardFont****Syntax**

```
FPD_Font FPDDocAddStandardFont (
    FPD_Document doc,
    const FS_CHAR* font,
    FPD_FontEncoding encoding
);
```

**Description**

Adds a Type1 base-14 font to the PDF document.

**Parameter**

---

doc	[In] A PDF document.
-----	----------------------

---

---

font	[In] The font name.
------	---------------------

---

---

encoding	[In] The font encoding.
----------	-------------------------

---

**Return****Head file reference**

fpd\_docTempl.h: 343

**FPDDocAddWindowsFont****Syntax**

```
FPD_Font FPDDocAddWindowsFont (
    FPD_Document doc,
    void* pLogFont,
    FS_BOOL bVert,
    FS_BOOL bTranslateName
);
```

**Description**

Imports a Windows font into the PDF document.

**Parameter**


---

doc	[In] A PDF document.
pLogFont	[In] Points to a LOGFONT(WIN32) structure that defines the characteristics of the logical font.
bVert	[In] Whether the font is a vertical font or not.
bTranslateName	[In] Whether we will do font name translating or not.

---

**Return**

A PDF font.

**Head file reference**

fpd\_docTempl.h: 319

**FPDDocAddWindowsFontW****Syntax**

```
FPD_Font FPDDocAddWindowsFontW (
    FPD Document doc,
    void* pLogFont,
    FS\_BOOL bVert,
    FS\_BOOL bTranslateName
);
```

**Description**

Imports a Windows font into the PDF document.

**Parameter**


---

doc	[In] A PDF document.
pLogFont	[In] Points to a LOGFONTW(WIN32) structure that defines the characteristics of the logical font.
bVert	[In] Whether the font is a vertical font or not.
bTranslateName	[In] Whether we will do font name translating or not.

---

**Return**

A PDF font.



**Head file reference**

fpd\_docTempl.h: 331

**FPDDocBuildResourceList****Syntax**

```
void FPDDocBuildResourceList (
    FPD_Document doc,
    const FS_CHAR* type,
    FS_PtrArray arr
);
```

**Description**

Build a list of all resources of specified type in this document. On return, the array contains pointers to FPD\_Object objects.

**Parameter**

---

doc	[In] A PDF document.
type	[In] The name of specified type.
arr	[Out] It will receive all resource objects of specified type in the document.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 354

**FPDDocCheckSignature****Syntax**

```
FS_BOOL FPDDocCheckSignature (
    FPD_Document doc
);
```

**Description**

Checks whether the document is signed or not.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

TRUE if the document is signed, otherwise not.

**Head file reference**

fpd\_docTempl.h: 608

**Since**

SDK LATEST VERSION > 2.1

## FPDDocClearModified

**Syntax**

```
void FPDDocClearModified (
    FPD_Document doc
);
```

**Description**

Clears the modified flag.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 578

## FPDDocClose

**Syntax**

```
FS_RESULT FPDDocClose (
    FPD_Document doc
);
```

**Description**

Closes the document returned by [FPDDocOpen \(\)](#), [FPDDocStartProgressiveLoad \(\)](#) or [FPDDocOpenMemDocument \(\)](#).

**Parameter**

---

doc	[In] The document returned by <a href="#">FPDDocOpen ()</a> , <a href="#">FPDDocStartProgressiveLoad ()</a> or <a href="#">FPDDocOpenMemDocument ()</a> .
-----	---

---

**Return**

Error code. FPD\_LOADERR\_SUCCESS for success.

**Head file reference**

fpd\_docTempl.h: 42

**Note:** Call FPDDocDestroy() to release a document which created by FPDDocNew().

**FPDDocContinueLoad****Syntax**

```
FS_RESULT FPDDocContinueLoad (
    FPD_Document doc,
    FS_PauseHandler pause
);
```

**Description**

Continue loading a PDF document. You must call [FPDDocClose](#) () once for every successful open.

**Parameter**

---

doc	[In] The PDF document.
pause	[In] The pause handler. This can be <a href="#">NULL</a> if you don't want to pause.

---

**Return**

The result code. See [FPDLoadErrCodes](#) .

**Head file reference**

fpd\_docTempl.h: 69

**FPDDocConvertIndirectObjects****Syntax**

```
void FPDDocConvertIndirectObjects (
    FPD_Document doc,
    FPD_Object obj,
    FS_BOOL bConvertStream,
    FS_BOOL bConvertDictionary
);
```

**Description**

Converts stream of dictionary members of an object to be indirect objects.

**Parameter**

---

doc	[In] A PDF document.
obj	[In] The input PDF object.
bConvertStream	[In] Whether we will convert stream to indirect object or not.
bConvertDictionary	[In] Whether we will convert dictionary to indirect object or not.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 386

**FPDDocCreateNewPage****Syntax**

```
FPD_Object FPDDocCreateNewPage (
    FPD_Document doc,
    FS_INT32 iPage
);
```

**Description**

Creates a new page in the PDF document. Return the created page.

**Parameter**


---

doc	[In] A PDF document.
iPage	[In] Specifies the zero-based index of page to be created.

---

**Return**

The created page dictionary.

**Head file reference**

fpd\_docTempl.h: 366

**FPDDocDeleteEnumPageHandler****Syntax**

```
void FPDDocDeleteEnumPageHandler (
    FPD_EnumPageHandler pEnumPageHandler
);
```

**Description**

Deletes the user-supplied page enumeration handler.

**Parameter**

---

pEnumPageHandler	[In] The user-supplied page enumeration handler.
------------------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 277

**FPDDocDeleteIndirectObject****Syntax**

```
void FPDDocDeleteIndirectObject (
    FPD\_Document doc,
    FS\_DWORD objNum
);
```

**Description**

Deletes an indirect object. Use this function with caution!

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

---

objNum	[In] The object number to delete.
--------	-----------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 479

**FPDDocDeletePage****Syntax**

```
void FPDDocDeletePage (
    FPD\_Document doc,
    FS\_INT32 iPage
);
```

**Description**

Deletes specified page in the PDF document.

**Parameter**

---

doc	[In] A PDF document.
-----	----------------------

---

iPage	[In] Specifies the zero-based index of page to be deleted.
-------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 376

**FPDDocDestroy****Syntax**

```
void FPDDocDestroy (
    FPD Document doc
);
```

**Description**

Destroys the document created by [FPDDocNew](#) . When Foxit Reader opens the doc, it is taken over and don't destroy it.

**Parameter**

---

doc	[In] The document to destroy.
-----	-------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 31

**FPDDocEnumPages****Syntax**

```
void FPDDocEnumPages (
    FPD Document doc,
    FPD EnumPageHandler pEnumPageHandler
);
```

**Description**

Enumerates pages with user-supplied page enumeration handler.

**Parameter**


---

doc	[In] A PDF document.
-----	----------------------

---

pEnumPageHandler	[In] The user-supplied page enumeration handler.
------------------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 258

**FPDDocGenerateFileID****Syntax**

```
FS_BOOL FPDDocGenerateFileID (
    FS_DWORD dwSeed1,
    FS_DWORD dwSeed2,
    FS_LPDWORD pBuffer
);
```

**Description**

Generates the ID of the PDF document.

**Parameter**


---

dwSeed1	[In] A seed value to initialize MT generator.
---------	---

---

dwSeed2	[In] A seed value to initialize MT generator.
---------	---

---

pBuffer	[Out] It receives the file ID.
---------	--------------------------------

---

**Return**TRUE for success, otherwise failure.**Head file reference**

fpd\_docTempl.h: 618

**Since**SDK LATEEST VERSION > 2.1.0.4**FPDDocGetID****Syntax**

```
void FPDDocGetID (
```

```
FPD_Document doc,
FS_ByteString* outID1,
FS_ByteString* outID2
);
```

**Description**

Gets ID strings of the document. Could be empty.

**Parameter**

doc	[In] A PDF document.
outID1	[Out] It receives the first ID string of the document.
outID2	[Out] It receives the second ID string of the document.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 119

**FPDDocGetIndirectObject****Syntax**

```
FPD_Object FPDDocGetIndirectObject (
    FPD_Document doc,
    FS_DWORD objNum
);
```

**Description**

Loads an indirect object by an object number.

**Parameter**

doc	[In] The input document.
objNum	[In] The input object number.

**Return**

An indirect object.

**Head file reference**

fpd\_docTempl.h: 438

## FPDDocGetIndirectObjsCount

### Syntax

```
FS_INT32 FPDDocGetIndirectObjsCount (
    FPD Document doc
);
```

### Description

Gets the indirect object count.

### Parameter

---

doc	[In] The input PDF document.
-----	------------------------------

---

### Return

The indirect object count.

### Head file reference

fpd\_docTempl.h: 673

### Since

[SDK LATEEST VERSION > 7.3.1.0](#)

## FPDDocGetIndirectType

### Syntax

```
FS_INT32 FPDDocGetIndirectType (
    FPD Document doc,
    FS DWORD objNum
);
```

### Description

Gets type of an document, without loading the object content.

### Parameter

---

doc	[In] The input document.
-----	--------------------------

---

---

objNum	[In] The input object number.
--------	-------------------------------

---

### Return

The type of the document.

### Head file reference

fpd\_docTempl.h: 448

## FPDDocGetInfo

### Syntax

```
FPD_Object FPDDocGetInfo (
    FPD Document doc
);
```

### Description

Gets document info dictionary from a PDF document. Could be NULL.

### Parameter

---

doc	[In] A PDF document.
-----	----------------------

---

### Return

The document info dictionary.

### Head file reference

fpd\_docTempl.h: 110

## FPDDocGetInfoObjNum

### Syntax

```
FS_DWORD FPDDocGetInfoObjNum (
    FPD Document doc
);
```

### Description

Gets the number of info object.

### Parameter

---

doc	[In] The input PDF document.
-----	------------------------------

---

### Return

The number of info object.

### Head file reference

fpd\_docTempl.h: 240

## FPDDocGetLastobjNum

### Syntax

```
FS_DWORD FPDDocGetLastobjNum (
    FPD Document doc
);
```

**Description**

Gets last assigned indirect object number.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

The number of the last object.

**Head file reference**

fpd\_docTempl.h: 530

**FPDDocGetNextAssoc****Syntax**

```
void FPDDocGetNextAssoc (
    FPD\_Document doc,
    FS\_POSITION* outPos,
    FS\_DWORD* outObjNum,
    FPD\_Object* outObject
);
```

**Description**

Accesses the indirect object of current position, and move the position to next.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

---

outPos	[In/Out] Input current position and receive the next position.
--------	--

---

---

outObjNum	[Out] It receives the current object number.
-----------	--

---

---

outObject	[Out] It receives the current indirect object pointer.
-----------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 557

**FPDDocGetPage**

**Syntax**

```
FPD_Object FPDDocGetPage (
    FPD Document doc,
    FS\_INT32 page_index
);
```

**Description**

Gets page dictionary of a specified page of the document. Page index starts from zero for the first page.

**Parameter**

---

doc	[In] A PDF document.
-----	----------------------

---

page_index	[In] Specifies the zero-based index of the page in the document.
------------	--

---

**Return**

The specified page dictionary of the document.

**Head file reference**

fpd\_docTempl.h: 130

**FPDDocGetPageContentModify****Syntax**

```
FPD_Object FPDDocGetPageContentModify (
    FPD Document doc,
    FPD Object page_dict
);
```

**Description**

Gets a modifiable content stream for a page in the PDF document.

**Parameter**

---

doc	[In] A PDF document.
-----	----------------------

---

page_dict	[In] The input page dictionary.
-----------	---------------------------------

---

**Return**

An modifiable content stream for the page.

**Head file reference**

fpd\_docTempl.h: 398

## FPDDocGetPageCount

### Syntax

```
FS_INT32 FPDDocGetPageCount (
    FPD Document doc
);
```

### Description

Gets number of pages in this document.

### Parameter

---

doc	[In]
-----	------

---

### Return

The page count.

### Head file reference

fpd\_docTempl.h: 130

## FPDDocGetPageIndex

### Syntax

```
FS_INT32 FPDDocGetPageIndex (
    FPD Document doc,
    FS DWORD objNum
);
```

### Description

Gets the page index from a indirect object number in the document.

### Parameter

---

doc	[In] A PDF document.
-----	----------------------

---

objNum	[In] The input indirect object number int the document.
--------	---

---

### Return

The zero-based index of page in the document.

### Head file reference

fpd\_docTempl.h: 149

## FPDDocGetParser

### Syntax



```
FPD_Parser FPDDocGetParser (
    FPD Document doc
);
```

**Description**

Gets the PDF file parser associated with a [FPD Document](#) object.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

The PDF parser.

**Head file reference**

fpd\_docTempl.h: 599

**FPDDocGetReviewType****Syntax**

```
FS_INT32 FPDDocGetReviewType (
    FPD Document doc
);
```

**Description**

Gets the review type. 0:normal, 1:share, 2:email.

**Parameter**

---

doc	[In] The document.
-----	--------------------

---

**Return**

The review type. 0:normal, 1:share, 2:email.

**Head file reference**

fpd\_docTempl.h: 630

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FPDDocGetRoot****Syntax**

```
FPD_Object FPDDocGetRoot (
    FPD Document doc
);
```

**Description**

Gets root dictionary (document catalog dictionary) from a PDF document.

**Parameter**

---

doc	[In] A PDF document.
-----	----------------------

---

**Return**

The root dictionary (document catalog dictionary).

**Head file reference**

fpd\_docTempl.h: 101

**FPDDocGetRootObjNum****Syntax**

```
FS_DWORD FPDDocGetRootObjNum (
    FPD Document doc
);
```

**Description**

Gets the number of root object.

**Parameter**

---

doc	[In] The input PDF document.
-----	------------------------------

---

**Return**

The number of root object.

**Head file reference**

fpd\_docTempl.h: 249

**FPDDocGetStartPosition****Syntax**

```
FS_POSITION FPDDocGetStartPosition (
    FPD Document doc
);
```

**Description**

Gets the start position of the indirect objects.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

The start position of the indirect objects.

**Head file reference**

fpd\_docTempl.h: 548

**FPDDoc GetUserPermissions****Syntax**

```
FS_DWORD FPDDoc GetUserPermissions (
    FPD Document doc
);
```

**Description**

Gets the user permission of the document.

**Parameter**

---

doc	[In] The input PDF document.
-----	------------------------------

---

**Return**

The user permission of the document.

**Head file reference**

fpd\_docTempl.h: 159

**FPDDoc ImportExternalObject****Syntax**

```
FPD_Object FPDDoc ImportExternalObject (
    FPD Document doc,
    FPD Object obj,
    FS MapPtrToPtr mapping
);
```

**Description**

Imports an object from external object collection as a new document. If the external object refers to other external indirect objects, those indirect objects are also imported. The mapping from old object number to new object number is updated during the import process.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

---

obj	[In] The object in external object collection.
-----	--

---

mapping	[Out] It updates the mapping from old object number to new object number.
---------	---

---

**Return**

A new indirect object.

**Head file reference**

fpd\_docTempl.h: 503

**FPDDocImportIndirectObject****Syntax**

```
FPD_Object FPDDocImportIndirectObject (
    FPD\_Document doc,
    FS\_LPBYTE pBuffer,
    FS\_DWORD size
);
```

**Description**

Imports an object from its binary format. This is used when the PDF is fetched in "Progressive Downloading" fashion. After this function call, the data buffer can be destroyed. This object must not be encrypted.

**Parameter**


---

doc	[In] The input document.
-----	--------------------------

---



---

pBuffer	[In] The binary data for the document. It must be not encrypted.
---------	--

---



---

size	[In] The size in bytes of the binary data.
------	--

---

**Return**

An object.

**Head file reference**

fpd\_docTempl.h: 489

**FPDDocInsertDocumentAtPos****Syntax**

```
FS_BOOL FPDDocInsertDocumentAtPos (
    FPD\_Document doc,
    FS\_INT32 index,
    FPD\_Document insertedDoc,
    const FS\_WCHAR\* pBookmarkTitle
```



```
 );
```

**Description**

Inserts the document into the original document at the specified position.

**Parameter**

doc	[In] The input document.
index	[In] The page index where the document will be inserted.
insertedDoc	[In] The inserted document.
pBookmarkTitle	[In] The input bookmark title. Set NULL as default.

**Return**

TRUE means success, otherwise failed.

**Head file reference**

fpd\_docTempl.h: 696

**Since**

[SDK\\_LATEEST\\_VERSION > 9.4.0](#)

**FPDDocInsertIndirectObject****Syntax**

```
void FPDDocInsertIndirectObject (
    FPD Document doc,
    FS\_DWORD objNum,
    FPD Object obj
);
```

**Description**

Inserts an indirect object with specified object number. This is used when the PDF is fetched in "Progressive Downloading" fashion, or parsing FDF. If an indirect object with the same object number exists, the previous one will be destroyed.

**Parameter**

doc	[In] The input document.
objNum	[In] The new object number of the indirect object to insert.
obj	[In] The indirect object to insert.



**Return**

void

**Head file reference**

fpd\_docTempl.h: 517

**FPDDocIsFormStream****Syntax**

```
FS_BOOL FPDDocIsFormStream (
    FPD Document doc,
    FS DWORD objnum,
    FS BOOL* outIsFormStream
);
```

**Description**

Checks if an indirect object is a form stream or not, without actually loading the object.

**Parameter**

---

doc	[In] A PDF document.
-----	----------------------

---

objnum	[In] The input indirect object number.
--------	--

---

outIsFormStream	[Out] It receive whether it's a form stream or not.
-----------------	---

---

**Return**

Non-zero means determined, otherwise unknown.

**Head file reference**

fpd\_docTempl.h: 177

**FPDDocIsModified****Syntax**

```
FS_BOOL FPDDocIsModified (
    FPD Document doc
);
```

**Description**

Checks if any of the indirect objects are modified, since loading from parser, or last ClearModified.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

TRUE if the document is modified, otherwise not.

**Head file reference**

fpd\_docTempl.h: 569

**FPDDocIsOwner****Syntax**

```
FS_BOOL FPDDocIsOwner (
    FPD Document doc
);
```

**Description**

Whether the user has the owner permission of the document.

**Parameter**

---

doc	[In] The input PDF document.
-----	------------------------------

---

**Return**

TRUE for being owner.

**Head file reference**

fpd\_docTempl.h: 168

**FPDDocIsPortfolio****Syntax**

```
FS_BOOL FPDDocIsPortfolio (
    FPD Document doc
);
```

**Description**

Checks whether the document is a portfolio document or not.

**Parameter**

---

doc	[In] The input PDF document.
-----	------------------------------

---

**Return**

TRUE if the document is a portfolio document, otherwise not.

**Head file reference**

fpd\_docTempl.h: 640

**Since**

[SDK LATEST VERSION > 2.1.0.4](#)

## FPDDocLoadColorSpace

**Syntax**

```
FPD_ColorSpace FPDDocLoadColorSpace (
    FPD\_Document doc,
    FPD\_Object csObj
);
```

**Description**

Loads a color space from a PDF object in the document.

**Parameter**

---

doc	[In] A PDF document.
-----	----------------------

---

csObj	[In] The input PDF object.
-------	----------------------------

---

**Return**

A PDF color space.

**Head file reference**

fpd\_docTempl.h: 198

## FPDDocLoadDoc

**Syntax**

```
void FPDDocLoadDoc (
    FPD\_Document doc
);
```

**Description**

Internally used

**Parameter**

---

doc	[In] The input PDF document.
-----	------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 429

## FPDDocLoadFont

### Syntax

```
FPD_Font FPDDocLoadFont (
    FPD\_Document doc,
    FPD\_Object fontDict
);
```

### Description

Loads a font from font dictionary in the document.

### Parameter

---

doc	[In] A PDF document.
-----	----------------------

---

fontDict	[In] The input font dictionary.
----------	---------------------------------

---

### Return

A PDF font.

### Head file reference

fpd\_docTempl.h: 188

## FPDDocLoadFontFile

### Syntax

```
FPD_StreamAcc FPDDocLoadFontFile (
    FPD\_Document doc,
    FPD\_Object stm
);
```

### Description

Loads a PDF stream accessor from a PDF stream in the document.

### Parameter

---

doc	[In] A PDF document.
-----	----------------------

---

stm	[In] The input PDF stream.
-----	----------------------------

---

### Return

A PDF stream accessor.



**Head file reference**

fpd\_docTempl.h: 230

**FPDDocLoadImageF****Syntax**

```
FPD_Image FPDDocLoadImageF (
    FPD\_Document doc,
    FPD\_Object obj,
    FS\_BOOL bCache
);
```

**Description**

Loads an image from a PDF object in the document.

**Parameter**

doc	[In] A PDF document.
obj	[In] The input PDF object.
bCache	[In] Whether we will cache the image or not.

**Return**

A PDF image.

**Head file reference**

fpd\_docTempl.h: 219

**FPDDocLoadPattern****Syntax**

```
FPD_Pattern FPDDocLoadPattern (
    FPD\_Document doc,
    FPD\_Object csObj,
    FS\_BOOL bShading
);
```

**Description**

Loads a pattern from a PDF object in the document.

**Parameter**

doc	[In] A PDF document.
csObj	[In] The input PDF object.

---

bShading	[In] Whether the pattern is a shading pattern or not.
----------	---

---

**Return**

A PDF pattern.

**Head file reference**

fpd\_docTempl.h: 208

**FPDDocNew****Syntax**

```
FPD_Document FPDDocNew (void );
```

**Description**

Creates a new empty document. The only [FPD Object](#) in the document will be a Catalog dictionary. See *Section 3.6 in the PDF Reference*.

**Return**

The new empty document.

**Head file reference**

fpd\_docTempl.h: 21

**FPDDocNewEnumPageHandler****Syntax**

```
FPD_EnumPageHandler FPDDocNewEnumPageHandler (
    FPD EnumPage enumPage
);
```

**Description**

Creates user-supplied page enumeration handler.

**Parameter**

---

enumPage	[In] The user-supplied page enumeration struct.
----------	---

---

**Return**

The user-supplied page enumeration handler.

**Head file reference**

fpd\_docTempl.h: 268

**FPDDocOpen****Syntax**

```
FPD_Document FPDDocOpen (
    FS_LPCWSTR wszFilePath,
    FS_LPCSTR szPassword
);
```

**Description**

Creates a new document from an existing PDF file. You must call [FPDDocClose\(\)](#) once for every successful open.

**Parameter**

---

wszFilePath	[In] The input file full path name.
-------------	-------------------------------------

---

szPassword	[In] The input password string.
------------	---------------------------------

---

**Return**

The newly created document from an existing PDF file.

**Head file reference**

fpd\_docTempl.h: 41

**FPDDocOpenMemDocument****Syntax**

```
FPD_Document FPDDocOpenMemDocument (
    void* dataBuf,
    FS_INT32 size,
    FS_LPCSTR szPassword
);
```

**Description**

Creates a document from memory block. You must call [FPDDocClose\(\)](#) once for every successful open.

**Parameter**

---

dataBuf	[In] The input memory block.
---------	------------------------------

---

size	[In] The size in bytes of the memory block.
------	---

---

szPassword	[In] The input password string.
------------	---------------------------------

---

**Return**

The newly created document from memory block.

**Head file reference**

---

fpd\_docTempl.h: 52

## FPDDocQuickSearch

### Syntax

```
FS_BOOL FPDDocQuickSearch (
    FPD\_Document doc,
    FS\_INT32 page_index,
    FS\_LPCWSTR pattern,
    FS\_BOOL bCaseSensitive
);
```

### Description

Quick search an pattern for specified page in the PDF document.

### Parameter

doc	[In] The input PDF document.
page_index	[In] The zero-based page index to be searched.
pattern	[In] The pattern to search.
bCaseSensitive	[In] Whether the pattern matching is case sensitive.

### Return

Non-zero means searched one successfully.

### Head file reference

fpd\_docTempl.h: 408

## FPDDocReleaseIndirectObject

### Syntax

```
void FPDDocReleaseIndirectObject (
    FPD\_Document doc,
    FS\_DWORD objNum
);
```

### Description

Sometimes, for saving memory space, we can release a loaded indirect object. However, use this with caution because the object pointer will become invalid.

### Parameter

---

doc	[In] The input document.
-----	--------------------------

---

objNum	[In] The object number to release.
--------	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 468

**FPDDocReloadFileStreams****Syntax**

```
void FPDDocReloadFileStreams (
    FPD\_Document doc
);
```

**Description**

Reload all file based stream when we do reparsing.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 539

**FPDDocReloadPages****Syntax**

```
void FPDDocReloadPages (
    FPD\_Document doc
);
```

**Description**

Reload page list. This can be used when document is progressively downloaded.

**Parameter**

---

doc	[In] A PDF document.
-----	----------------------

---

**Return**

void



**Head file reference**

fpd\_docTempl.h: 420

**FPDDocRemoveUR3****Syntax**

```
FS_BOOL FPDDocRemoveUR3 (
    FPD Document doc
);
```

**Description**

Removes the UR3 signature.

**Parameter**

---

doc	[In] The input PDF document.
-----	------------------------------

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 650

**Since**[SDK LATEEST VERSION > 7.1.0.0](#)**FPDDocSave****Syntax**

```
FS_BOOL FPDDocSave (
    FPD Document doc,
    FS LPCSTR filename,
    FS DWORD flags,
    FS BOOL bEnable
);
```

**Description**

Writes the whole document into a new file.

**Parameter**

---

doc	[In] The input document.
-----	--------------------------

---

filename	[In] The output file name.
----------	----------------------------

---



---

flags	[In] The saving flags. You can set FPD_SAVE_DEFAULT by default.
-------	---

---

bEnable	[In] Whether to do data compressing.
---------	--------------------------------------

---

**Return**

Non-zero means success, otherwise failed.

**Head file reference**

fpd\_docTempl.h: 587

**FPDDocSave2****Syntax**

```
FS_BOOL FPDDocSave2 (
    FPD Document doc,
    FS LPCWSTR wStrfilename,
    FS DWORD flags,
    FS BOOL bEnable
);
```

**Description**

Writes the whole document into a new file.

**Parameter**


---

doc	[In] The input document.
-----	--------------------------

---

wStrfilename	[In] The output file name,wide string type.
--------------	---

---

flags	[In] The saving flags. You can set FPD_SAVE_DEFAULT by default.
-------	---

---

bEnable	[In] Whether to do data compressing.
---------	--------------------------------------

---

**Return**

Non-zero means success, otherwise failed.

**Head file reference**

fpd\_docTempl.h: 683

**Since**

[SDK LATEEST VERSION > 8.3.1](#)

**FPDDocSetID**

**Syntax**

```
void FPDDocSetID (
    FPD Document doc,
    FS\_LPCBYTE szID1,
    FS\_INT32 nLen1,
    FS\_LPCBYTE szID2,
    FS\_INT32 nLen2
);
```

**Description**

Sets the PDF file ID.

**Parameter**

---

doc	[In] A PDF document.
-----	----------------------

---

szID1	[In] The first file ID.
-------	-------------------------

---

nLen1	[In] The length of the first ID byte string.
-------	--

---

szID2	[In] The second file ID.
-------	--------------------------

---

nLen2	[In] The length of the second ID byte string.
-------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 306

**FPDDocSetInfoObjNum****Syntax**

```
void FPDDocSetInfoObjNum (
    FPD Document doc,
    FS\_INT32 InfoObjNum
);
```

**Description**

Sets the info object number in the PDF document. Must be called for an empty document.

**Parameter**

---

doc	[In] A empty PDF document.
-----	----------------------------

---



---

InfoObjNum	[In] The input info object number.
------------	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 296

**FPDDocSetRootObjNum****Syntax**

```
void FPDDocSetRootObjNum (
    FPD\_Document doc,
    FS\_INT32 RootObjNum
);
```

**Description**

Sets the root object number in the PDF document. Must be called for an empty document.

**Parameter**


---

doc	[In] A empty PDF document.
-----	----------------------------

---



---

RootObjNum	[In] The input root object number.
------------	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 286

**FPDDocStartProgressiveLoad****Syntax**

```
FS_RESULT FPDDocStartProgressiveLoad (
    FS\_FileReadHandler fileRead,
    FS\_LPCSTR szPassword,
    FS\_PauseHandler pause,
    FPD\_Document* outPDDoc
);
```

**Description**

Document loading is a progressive process. It might take a long time to load a document, especially when a file is corrupted, FPDFEMB will try to recover the document contents by scanning the whole file. If "pause" parameter is provided, this function may return FPDFERR\_TOBECONTINUED any time during the document loading.

When [FPD\\_LOADERR\\_TOBECONTINUED](#) is returned, the "outPDDoc" parameter will still receive a valid document handle, however, no further operations can be performed on



the document, except the [FPDDocContinueLoad\(\)](#) function call, which resume the document loading.

#### Parameter

---

fileRead	[In] The file access handler. Plug-ins should free the object after the document is parsed completely.
szPassword	[In] The input password string.
pause	[In] The pause handler. This can be <a href="#">NULL</a> if you don't want to pause.
outPDDoc	[Out] Receiving the document.

---

#### Return

The result code. See [FPDLoadErrCodes](#).

#### Head file reference

fpd\_docTempl.h: 64

## FPD\_EnumPageHandler

#### [Return from Used by](#)

#### Description

A handler to enumerate the count of pages for a document. See [FPDDocNewEnumPageHandler](#), [FPDDocDeleteEnumPageHandler](#).

#### Returned from

#### [FPDDocNewEnumPageHandler](#)

#### Used by

#### [FPDDocDeleteEnumPageHandler](#) [FPDDocEnumPages](#)

#### Callbacks

##### Callbacks summary

##### [EnumPage](#)

Enumerates page function for call back.

##### Callbacks detail



## EnumPage

### Syntax

```
typedef FS_BOOL (*EnumPage)(  
    FS_LPVVOID clientData,  
    FPD_Object pageDict  
>;
```

### Description

Enumerates page function for call back. Enumerates page function for call back. Returns [TRUE](#) if you want the enumeration to continue.

### Parameter

clientData	[In] The user-supplied data.
------------	------------------------------

pageDict	[In] The page dictionary.
----------	---------------------------

### Return

Non-zero means you want the enumeration to continue, otherwise want not.

### Head file reference

fpd\_docExpT.h: 1328

### Group

[FPD\\_EnumPage](#)

## FPD\_EPUB

### Description

## FPD\_FileSpec

### [Return from Used by](#)

### Description

PDF file specification object. A FPD\_FileSpec corresponds to the PDF file specification object (see Section 3.10, File Specifications, in the PDF Reference). It is used to specify a file in an action.

### Returned from

[FPDFFileSpecNew](#)  
[FPDFFileSpecNewFromObj](#)

### Used by



[FPDRenditionGetMediaClipFile](#)  
[FPDRenditionSetMediaClipFile](#)  
[FPDFFileSpecDestroy](#)  
[FPDFFileSpec.GetFileName](#)  
[FPDFFileSpecGetFileStream](#)  
[FPDFFileSpecIsURL](#)  
[FPDFFileSpecSetEmbeddedFile](#)  
[FPDFFileSpecSetFileName](#)

## Functions

### Functions summary

#### [FPDFFileSpecDestroy](#)

Destroys a file specification object.

#### [FPDFFileSpec.GetFileName](#)

Gets the file name.

#### [FPDFFileSpecGetFileStream](#)

Gets a PDF stream from the file specification.

#### [FPDFFileSpecIsURL](#)

Checks whether it's a URL or not. If is an URL, FPDFFileSpec.GetFileName gets the URL address.

#### [FPDFFileSpecNew](#)

Creates a new file specification object - dictionary object.

#### [FPDFFileSpecNewFromObj](#)

Creates a new file specification object from a PDF object.

#### [FPDFFileSpecSetEmbeddedFile](#)

Sets an embedded file.

#### [FPDFFileSpecSetFileName](#)

Sets the file name.

### Functions detail

#### FPDFFileSpecDestroy

##### Syntax

```
void FPDFFileSpecDestroy (
    FPD\_FileSpec fileSpec
);
```

##### Description

Destroys a file specification object.

##### Parameter

---

fileSpec	[In] A file specification object.
----------	-----------------------------------

---

##### Return

void

##### Head file reference



fpd\_docTempl.h: 2474

## FPDFFileSpec.GetFileName

### Syntax

```
FS_BOOL FPDFFileSpec.GetFileName (
    FPD\_FileSpec fileSpec,
    FS\_WideString* outFile
);
```

### Description

Gets the file name.

### Parameter

---

fileSpec	[In] A file specification object.
----------	-----------------------------------

---

outFile	[Out] It receives the file name.
---------	----------------------------------

---

### Return

Non-zero means success, otherwise failure.

### Head file reference

fpd\_docTempl.h: 2485

## FPDFFileSpec.GetFileStream

### Syntax

```
FPD_Object FPDFFileSpec.GetFileStream (
    FPD\_FileSpec fileSpec
);
```

### Description

Gets a PDF stream from the file specification.

### Parameter

---

fileSpec	[In] A file specification object.
----------	-----------------------------------

---

### Return

The PDF stream object from the file specification.

### Head file reference

fpd\_docTempl.h: 2503



## FPDFFileSpecIsURL

### Syntax

```
FS_BOOL FPDFFileSpecIsURL (  
    FPD\_FileSpec fileSpec  
) ;
```

### Description

Checks whether it's a URL or not. If is an URL, FPDFFileSpecGetFileName gets the URL address.

### Parameter

---

fileSpec	[In] A file specification object.
----------	-----------------------------------

---

### Return

[TRUE](#) for being a URL, otherwise not.

### Head file reference

fpd\_docTempl.h: 2483

## FPDFFileSpecNew

### Syntax

```
FPD_FileSpec FPDFFileSpecNew (void );
```

### Description

Creates a new file specification object - dictionary object.

### Return

A new file specification object.

### Head file reference

fpd\_docTempl.h: 2456

## FPDFFileSpecNewFromObj

### Syntax

```
FPD_FileSpec FPDFFileSpecNewFromObj (  
    FPD\_Object obj  
) ;
```

### Description

Creates a new file specification object from a PDF object.

### Parameter



---

obj	[In] The input PDF object.
-----	----------------------------

---

**Return**

A new file specification object.

**Head file reference**

fpd\_docTempl.h: 2465

**FPDFFileSpecSetEmbeddedFile****Syntax**

```
void FPDFFileSpecSetEmbeddedFile (
    FPD\_FileSpec fileSpec,
    FPD\_Document doc,
    FS\_FileReadHandler pFileReadHandler,
    FS\_LPCWSTR szFilePath
);
```

**Description**

Sets an embedded file.

**Parameter**


---

fileSpec	[In] A file specification object.
----------	-----------------------------------

---



---

doc	[In] The PDF document.
-----	------------------------

---



---

pFileReadHandler	[In] The file access interface.
------------------	---------------------------------

---



---

szFilePath	[In] The file path.
------------	---------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2523

**FPDFFileSpecSetFileName****Syntax**

```
void FPDFFileSpecSetFileName (
    FPD\_FileSpec fileSpec,
    FS\_LPCWSTR wszFile,
    FS\_BOOL bURL
);
```



**Description**

Sets the file name.

**Parameter**

fileSpec	[In] A file specification object.
wszFile	[In] Input a file name.
bURL	[In] Whether the input file name is an URL.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2512

## FPD\_Font

**[Return from Used by](#)****Description**

The [FPD\\_Font](#) object is used to manipulate the PDF font. See [FPDFFontNew](#) , [FPDFFontDestroy](#) .

**Returned from**

[FPDCIDFontNew](#)  
[FPDDocAddFXFont](#)  
[FPDDocAddStandardFont](#)  
[FPDDocAddWindowsFont](#)  
[FPDDocAddWindowsFontW](#)  
[FPDDocLoadFont](#)  
[FPDFFontEncodingNew](#)  
[FPDFFontEncodingNew2](#)  
[FPDFFormControlGetDefaultControlFont](#)  
[FPDFFormFieldGetFont](#)  
[FPDIFormAddNativeFont](#)  
[FPDIFormAddNativeFont2](#)  
[FPDIFormAddNativeFormFont](#)  
[FPDIFormAddNativeFormFont2](#)  
[FPDIFormAddStandardFont](#)  
[FPDIFormAddSystemDefaultFont](#)  
[FPDIFormAddSystemFont](#)  
[FPDIFormAddSystemFontW](#)  
[FPDIFormGetDefaultFormFont](#)  
[FPDIFormGetFormFont](#)



[FPDInterFormGetFormFont2](#)  
[FPDInterFormGetFormFont3](#)  
[FPDInterFormGetNativeFormFont](#)  
[FPDInterFormGetNativeFormFont2](#)  
[FPDTextObjectGetFont](#)  
[FPDTextStateGetFont](#)  
[FPDTrueTypeFontGetEncoding](#)  
[FPDTrueTypeFontNew](#)  
[FPDType1FontGetEncoding](#)  
[FPDType1FontNew](#)  
[FPDType3FontGetEncoding](#)  
[FPDType3FontNew](#)  
[FPDFontGetCIDFont](#)  
[FPDFontGetStockFont](#)  
[FPDFontGetTrueTypeFont](#)  
[FPDFontGetType1Font](#)  
[FPDFontGetType3Font](#)  
[FPDFontNew](#)

### Used by

[FPDCIDFontAppendChar](#)  
[FPDCIDFontCIDFromCharCode](#)  
[FPDCIDFontCountChar](#)  
[FPDCIDFontDestroy](#)  
[FPDCIDFontGetCharBBox](#)  
[FPDCIDFontGetCharset](#)  
[FPDCIDFontGetCharSize](#)  
[FPDCIDFontGetCharWidthF](#)  
[FPDCIDFontGetCIDTransform](#)  
[FPDCIDFontGetNextChar](#)  
[FPDCIDFontGetVertOrigin](#)  
[FPDCIDFontGetVertWidth](#)  
[FPDCIDFontGlyphFromCharCode](#)  
[FPDCIDFontIsTrueType](#)  
[FPDCIDFontIsUnicodeCompatible](#)  
[FPDCIDFontIsVertWriting](#)  
[FPDCIDFontLoadGB2312](#)  
[FPDDocAddStandardFont](#)  
[FPDFontEncodingCharCodeFromUnicode](#)  
[FPDFontEncodingDestroy](#)  
[FPDFontEncodingIsIdentical](#)  
[FPDFontEncodingLoadEncoding](#)  
[FPDFontEncodingRealize](#)  
[FPDFontEncodingSetUnicode](#)  
[FPDFontEncodingUnicodeFromCharCode](#)  
[FPDFFormFindFontName](#)  
[FPDFXFontEncodingNew](#)  
[FPDInterFormFindFormFont2](#)  
[FPDInterFormFindFormFont3](#)  
[FPDPageFindFontName](#)  
[FPDRenderDeviceDrawNormalText](#)  
[FPDRenderDeviceDrawTextPath](#)  
[FPDRenderDeviceDrawTextString](#)  
[FPDRenderDeviceDrawTextString2](#)  
[FPDRenderDeviceDrawType3Text](#)

[FPDTextStateSetFont](#)  
[FPDTrueTypeFontDestroy](#)  
[FPDTrueTypeFontGetCharBBox](#)  
[FPDTrueTypeFontGetCharWidthF](#)  
[FPDTrueTypeFontGetEncoding](#)  
[FPDTrueTypeFontGlyphFromCharCode](#)  
[FPDTrueTypeFontIsUnicodeCompatible](#)  
[FPDType1FontDestroy](#)  
[FPDType1FontGetBase14Font](#)  
[FPDType1FontGetCharBBox](#)  
[FPDType1FontGetCharWidthF](#)  
[FPDType1FontGetEncoding](#)  
[FPDType1FontGlyphFromCharCode](#)  
[FPDType1FontIsUnicodeCompatible](#)  
[FPDType3FontDestroy](#)  
[FPDType3FontGetCharBBox](#)  
[FPDType3FontGetCharTypeWidth](#)  
[FPDType3FontGetCharWidthF](#)  
[FPDType3FontGetEncoding](#)  
[FPDType3FontGetFontMatrix](#)  
[FPDType3FontGlyphFromCharCode](#)  
[FPDType3FontIsUnicodeCompatible](#)  
[FPDType3FontLoadChar](#)  
[FPDType3FontSetPageResources](#)  
[FPDFontAppendChar](#)  
[FPDFontAppendChar2](#)  
[FPDFontCharCodeFromUnicode](#)  
[FPDFontCountChar](#)  
[FPDFontDecodeString](#)  
[FPDFontDestroy](#)  
[FPDFontEncodeString](#)  
[FPDFontGetBaseFont](#)  
[FPDFontGetCharBBox](#)  
[FPDFontGetCharMap](#)  
[FPDFontGetCharSize](#)  
[FPDFontGetCharTypeWidth](#)  
[FPDFontGetCharWidthF](#)  
[FPDFontGetCIDFont](#)  
[FPDFontGetFace](#)  
[FPDFontGetFlags](#)  
[FPDFontGetFontBBox](#)  
[FPDFontGetFontDict](#)  
[FPDFontGetFontFile](#)  
[FPDFontGetFontType](#)  
[FPDFontGetFontTypeName](#)  
[FPDFontGetFXFont](#)  
[FPDFontGetItalicAngle](#)  
[FPDFontGetNextChar](#)  
[FPDFontGetPDFDoc](#)  
[FPDFontGetStringWidth](#)  
[FPDFontGetSubstFont](#)  
[FPDFontGetTrueTypeFont](#)  
[FPDFontGetType1Font](#)  
[FPDFontGetType3Font](#)  
[FPDFontGetTypeAscent](#)  
[FPDFontGetTypeDescent](#)  
[FPDFontGlyphFromCharCode](#)



[FPDFFontIsCharEmbedded](#)  
[FPDFFontIsEmbedded](#)  
[FPDFFontIsStandardFont](#)  
[FPDFFontIsUnicodeCompatible](#)  
[FPDFFontIsVertWriting](#)  
[FPDFFontLoadGlyphPath](#)  
[FPDFFontRecognizeChar](#)  
[FPDFFontUnicodeFromCharCode](#)  
[FPDFFontUnicodeFromCharCodeEx](#)

## Definitions

### Definitions summary

#### [FPD\\_CS\\_CALGRAY](#)

CalGray.

#### [FPD\\_CS\\_CALRGB](#)

CalRGB.

#### [FPD\\_CS\\_DEVICECMYK](#)

DeviceCMYK.

#### [FPD\\_CS\\_DEVICEGRAY](#)

DeviceGray.

#### [FPD\\_CS\\_DEVICEN](#)

DeviceN.

#### [FPD\\_CS\\_DEVICERGB](#)

DeviceRGB.

#### [FPD\\_CS\\_ICCBASED](#)

ICCBased.

#### [FPD\\_CS\\_INDEXED](#)

Indexed.

#### [FPD\\_CS\\_LAB](#)

Lab.

#### [FPD\\_CS\\_PATTERN](#)

Pattern.

#### [FPD\\_CS\\_SEPARATION](#)

Separation.

#### [FPD\\_FONT\\_ALLCAP](#)

Font contains no lowercase letters.

#### [FPD\\_FONT\\_FIXEDPITCH](#)

All glyphs have the same width.

#### [FPD\\_FONT\\_FORCEBOLD](#)

Whether bold glyphs are painted with extra pixels even at very small text sizes.

#### [FPD\\_FONT\\_ITALIC](#)

Glyphs have dominant vertical strokes that are slanted.

#### [FPD\\_FONT\\_NONSYMBOLIC](#)

Font uses the Adobe standard Latin character set or a subset of it.

#### [FPD\\_FONT\\_SCRIPT](#)

Glyphs resemble cursive handwriting.

#### [FPD\\_FONT\\_SERIF](#)

Glyphs have serifs, which are short strokes drawn at an angle on the top and bottom of glyph stems.



**FPD FONT SMALLCAP**

Font contains both uppercase and lowercase letters.

**FPD FONT SYMBOLIC**

Font contains glyphs outside the Adobe standard Latin character set.

**FPD FONT ENCODING ADOBE\_SYMBOL**

Adobe symbol encoding.

**FPD FONT ENCODING BUILTIN**

Built-in encoding.

**FPD FONT ENCODING MACEXPERT**

Mac expert encoding.

**FPD FONT ENCODING MACROMAN**

Mac roman encoding.

**FPD FONT ENCODING MS\_SYMBOL**

Microsoft symbol encoding.

**FPD FONT ENCODING PDFDOC**

PDF Document encoding.

**FPD FONT ENCODING STANDARD**

Adobe standard encoding.

**FPD FONT ENCODING UNICODE**

Unicode encoding.

**FPD FONT ENCODING WINANSI**

Windows ansic encoding.

**FPD FONT ENCODING ZAPFDINGBATS**

ZapfDingbats encoding.

**FPD FONT CIDFONT**

CID font.

**FPD FONT TRUETYPE**

True type.

**FPD FONT TYPE1**

Type1.

**FPD FONT TYPE3**

Type3.

## Definitions detail

### **FPD\_CS\_CALGRAY**

**Syntax**

```
#define FPD_CS_CALGRAY 4
```

**Description**

CalGray.

**Group**

[FPDFontColorSpaceFamilies](#)

**Head file reference**

fpd\_resourceExpT.h: 145



## FPD\_CS\_CALRGB

### Syntax

```
#define FPD_CS_CALRGB 5
```

### Description

CalRGB.

### Group

[FPDFFontColorSpaceFamilies](#)

### Head file reference

fpd\_resourceExpT.h: 147

## FPD\_CS\_DEVICECMYK

### Syntax

```
#define FPD_CS_DEVICECMYK 3
```

### Description

DeviceCMYK.

### Group

[FPDFFontColorSpaceFamilies](#)

### Head file reference

fpd\_resourceExpT.h: 143

## FPD\_CS\_DEVICEGRAY

### Syntax

```
#define FPD_CS_DEVICEGRAY 1
```

### Description

DeviceGray.

### Group

[FPDFFontColorSpaceFamilies](#)

### Head file reference

fpd\_resourceExpT.h: 139

## FPD\_CS\_DEVICEN

### Syntax

```
#define FPD_CS_DEVICEN 9
```



**Description**

DeviceN.

**Group**

[FPDFFontColorSpaceFamilies](#)

**Head file reference**

fpd\_resourceExpT.h: 155

## FPD\_CS\_DEVICERGB

**Syntax**

```
#define FPD_CS_DEVICERGB 2
```

**Description**

DeviceRGB.

**Group**

[FPDFFontColorSpaceFamilies](#)

**Head file reference**

fpd\_resourceExpT.h: 141

## FPD\_CS\_ICCBASED

**Syntax**

```
#define FPD_CS_ICCBASED 7
```

**Description**

ICCBased.

**Group**

[FPDFFontColorSpaceFamilies](#)

**Head file reference**

fpd\_resourceExpT.h: 151

## FPD\_CS\_INDEXED

**Syntax**

```
#define FPD_CS_INDEXED 10
```

**Description**

Indexed.

**Group**



[FPDFFontColorSpaceFamilies](#)

**Head file reference**

fpd\_resourceExpT.h: 157

## FPD\_CS\_LAB

**Syntax**

#define FPD\_CS\_LAB 6

**Description**

Lab.

**Group**

[FPDFFontColorSpaceFamilies](#)

**Head file reference**

fpd\_resourceExpT.h: 149

## FPD\_CS\_PATTERN

**Syntax**

#define FPD\_CS\_PATTERN 11

**Description**

Pattern.

**Group**

[FPDFFontColorSpaceFamilies](#)

**Head file reference**

fpd\_resourceExpT.h: 159

## FPD\_CS\_SEPARATION

**Syntax**

#define FPD\_CS\_SEPARATION 8

**Description**

Separation.

**Group**

[FPDFFontColorSpaceFamilies](#)

**Head file reference**

fpd\_resourceExpT.h: 153



## FPD\_FONT\_ALLCAP

### Syntax

```
#define FPD_FONT_ALLCAP 0x10000
```

### Description

Font contains no lowercase letters.

### Group

[FPDFontFlags](#)

### Head file reference

fpd\_resourceExpT.h: 93

## FPD\_FONT\_FIXEDPITCH

### Syntax

```
#define FPD_FONT_FIXEDPITCH 1
```

### Description

All glyphs have the same width.

### Group

[FPDFontFlags](#)

### Head file reference

fpd\_resourceExpT.h: 81

## FPD\_FONT\_FORCEBOLD

### Syntax

```
#define FPD_FONT_FORCEBOLD 0x40000
```

### Description

Whether bold glyphs are painted with extra pixels even at very small text sizes.

### Group

[FPDFontFlags](#)

### Head file reference

fpd\_resourceExpT.h: 97

## FPD\_FONT\_ITALIC

### Syntax

```
#define FPD_FONT_ITALIC 64
```



**Description**

Glyphs have dominant vertical strokes that are slanted.

**Group**

[FPDFontFlags](#)

**Head file reference**

fpd\_resourceExpT.h: 91

## FPD\_FONT\_NONSYMBOLIC

**Syntax**

#define FPD\_FONT\_NONSYMBOLIC 32

**Description**

Font uses the Adobe standard Latin character set or a subset of it.

**Group**

[FPDFontFlags](#)

**Head file reference**

fpd\_resourceExpT.h: 89

## FPD\_FONT\_SCRIPT

**Syntax**

#define FPD\_FONT\_SCRIPT 8

**Description**

Glyphs resemble cursive handwriting.

**Group**

[FPDFontFlags](#)

**Head file reference**

fpd\_resourceExpT.h: 87

## FPD\_FONT\_SERIF

**Syntax**

#define FPD\_FONT\_SERIF 2

**Description**

Glyphs have serifs, which are short strokes drawn at an angle on the top and bottom of glyph stems.



**Group**

[FPDFontFlags](#)

**Head file reference**

fpd\_resourceExpT.h: 83

## FPD\_FONT\_SMALLCAP

**Syntax**

#define FPD\_FONT\_SMALLCAP 0x20000

**Description**

Font contains both uppercase and lowercase letters.

**Group**

[FPDFontFlags](#)

**Head file reference**

fpd\_resourceExpT.h: 95

## FPD\_FONT\_SYMBOLIC

**Syntax**

#define FPD\_FONT\_SYMBOLIC 4

**Description**

Font contains glyphs outside the Adobe standard Latin character set.

**Group**

[FPDFontFlags](#)

**Head file reference**

fpd\_resourceExpT.h: 85

## FPD\_FONT\_ENCODING\_ADOBE\_SYMBOL

**Syntax**

#define FPD\_FONT\_ENCODING\_ADOBE\_SYMBOL 5

**Description**

Adobe symbol encoding.

**Group**

[FPDFontPredefinedEncoding](#)

**Head file reference**



fpd\_resourceExpT.h: 119

## FPD\_FONT\_ENCODING\_BUILTIN

### Syntax

#define FPD\_FONT\_ENCODING\_BUILTIN 0

### Description

Built-in encoding.

### Group

[FPDFontPredefinedEncoding](#)

### Head file reference

fpd\_resourceExpT.h: 109

## FPD\_FONT\_ENCODING\_MACEXPERT

### Syntax

#define FPD\_FONT\_ENCODING\_MACEXPERT 3

### Description

Mac expert encoding.

### Group

[FPDFontPredefinedEncoding](#)

### Head file reference

fpd\_resourceExpT.h: 115

## FPD\_FONT\_ENCODING\_MACROMAN

### Syntax

#define FPD\_FONT\_ENCODING\_MACROMAN 2

### Description

Mac roman encoding.

### Group

[FPDFontPredefinedEncoding](#)

### Head file reference

fpd\_resourceExpT.h: 113

## FPD\_FONT\_ENCODING\_MS\_SYMBOL



**Syntax**

```
#define FPD_FONT_ENCODING_MS_SYMBOL 8
```

**Description**

Microsoft symbol encoding.

**Group**

[FPDFontPredefinedEncoding](#)

**Head file reference**

fpd\_resourceExpT.h: 125

## FPD\_FONT\_ENCODING\_PDFDOC

**Syntax**

```
#define FPD_FONT_ENCODING_PDFDOC 7
```

**Description**

PDF Document encoding.

**Group**

[FPDFontPredefinedEncoding](#)

**Head file reference**

fpd\_resourceExpT.h: 123

## FPD\_FONT\_ENCODING\_STANDARD

**Syntax**

```
#define FPD_FONT_ENCODING_STANDARD 4
```

**Description**

Adobe standard encoding.

**Group**

[FPDFontPredefinedEncoding](#)

**Head file reference**

fpd\_resourceExpT.h: 117

## FPD\_FONT\_ENCODING\_UNICODE

**Syntax**

```
#define FPD_FONT_ENCODING_UNICODE 9
```

**Description**

Unicode encoding.



**Group**

[FPDFontPredefinedEncoding](#)

**Head file reference**

fpd\_resourceExpT.h: 127

## FPD\_FONT\_ENCODING\_WINANSI

**Syntax**

#define FPD\_FONT\_ENCODING\_WINANSI 1

**Description**

Windows ansic encoding.

**Group**

[FPDFontPredefinedEncoding](#)

**Head file reference**

fpd\_resourceExpT.h: 111

## FPD\_FONT\_ENCODING\_ZAPFDINGBATS

**Syntax**

#define FPD\_FONT\_ENCODING\_ZAPFDINGBATS 6

**Description**

ZapfDingbats encoding.

**Group**

[FPDFontPredefinedEncoding](#)

**Head file reference**

fpd\_resourceExpT.h: 121

## FPD\_FONT\_CIDFONT

**Syntax**

#define FPD\_FONT\_CIDFONT 4

**Description**

CID font.

**Group**

[FPDFontTypeIDs](#)

**Head file reference**



fpd\_resourceExpT.h: 68

## FPD\_FONT\_TRUETYPE

### Syntax

#define FPD\_FONT\_TRUETYPE 2

### Description

True type.

### Group

[FPDFFontTypeIDs](#)

### Head file reference

fpd\_resourceExpT.h: 64

## FPD\_FONT\_TYPE1

### Syntax

#define FPD\_FONT\_TYPE1 1

### Description

Type1.

### Group

[FPDFFontTypeIDs](#)

### Head file reference

fpd\_resourceExpT.h: 62

## FPD\_FONT\_TYPE3

### Syntax

#define FPD\_FONT\_TYPE3 3

### Description

Type3.

### Group

[FPDFFontTypeIDs](#)

### Head file reference

fpd\_resourceExpT.h: 66

## Functions



## Functions summary

### [\*\*FPDFFontAppendChar\*\*](#)

Append a charcode to a string buffer.

### [\*\*FPDFFontAppendChar2\*\*](#)

Append a charcode to a string buffer.

### [\*\*FPDFFontCharCodeFromUnicode\*\*](#)

Gets a charcode from a Unicode.

### [\*\*FPDFFontCountChar\*\*](#)

Gets the count of characters in a string.

### [\*\*FPDFFontDecodeString\*\*](#)

Decode a font string to unicode string.

### [\*\*FPDFFontDestroy\*\*](#)

Destroys the PDF font object.

### [\*\*FPDFFontEncodeString\*\*](#)

Encode an unicode string to font string.

### [\*\*FPDFFontFXFontGetFaceName\*\*](#)

Gets the face name of the Foxit GE font.

### [\*\*FPDFFontFXFontGetFamilyName\*\*](#)

Gets the family name of the Foxit GE font.

### [\*\*FPDFFontFXFontGetPsName\*\*](#)

Gets the postscript name of the Foxit GE font.

### [\*\*FPDFFontFXFontIsBold\*\*](#)

Checks whether it is bold font or not.

### [\*\*FPDFFontFXFontIsItalic\*\*](#)

Checks whether it is italic font or not.

### [\*\*FPDFFontFXFontLoadSubst\*\*](#)

Loads a font through font mapper.

### [\*\*FPDFFontGetBaseFont\*\*](#)

Gets the base font name.

### [\*\*FPDFFontGetCharBBox\*\*](#)

Gets the bounding box of a character.

### [\*\*FPDFFontGetCharMap\*\*](#)

Gets the character map.

### [\*\*FPDFFontGetCharSize\*\*](#)

Gets the number of bytes for the char code.

### [\*\*FPDFFontGetCharTypeWidth\*\*](#)

Gets a character's real width in the font file.

### [\*\*FPDFFontGetCharWidthF\*\*](#)

Gets a character's PDF width.

### [\*\*FPDFFontGetCIDFont\*\*](#)

Gets a CID font.

### [\*\*FPDFFontGetFace\*\*](#)

Gets embedded or substituted FT font face.

### [\*\*FPDFFontGetFlags\*\*](#)

Gets the font flags.

### [\*\*FPDFFontGetFontBBox\*\*](#)

Gets the font's bounding box.

### [\*\*FPDFFontGetFontDict\*\*](#)

Gets the font dictionary.



**FPDFFontGetFontFile**

Gets the font stream.

**FPDFFontGetFontType**

Gets the font type.

**FPDFFontGetFontTypeName**

Gets the font type name.

**FPDFFontGetFXFont**

Gets the Foxit GE font.

**FPDFFontGetItalicAngle**

Gets the angle, expressed in degrees counterclockwise from the vertical, of the dominant vertical strokes of the font.

**FPDFFontGetNextChar**

Gets a charcode from a string at specified position.

**FPDFFontGetPDFDoc**

Gets the PDF document.

**FPDFFontGetStockFont**

Gets the "stocked" fonts (Adobe base 14 fonts), which are always available for rendering activities. However, these fonts can't be directly referred in any document, unless the font resource is added to the document.

**FPDFFontGetStringWidth**

Gets a string's width.

**FPDFFontGetSubstFont**

Gets the substitute font.

**FPDFFontGetSubstFontCharset**

Gets the charset of the substitution font.

**FPDFFontGetSubstFontWeight**

Gets the weight parameter, non-zero for synthetic weight generation only.

**FPDFFontGetTrueTypeFont**

Gets a true type font.

**FPDFFontGetType1Font**

Gets a type1 font.

**FPDFFontGetType3Font**

Gets a type3 font.

**FPDFFontGetTypeAscent**

Gets the typographic ascent.

**FPDFFontGetTypeDescent**

Gets the typographic descent, most negative.

**FPDFFontGlyphFromCharCode**

Gets the glyph index for a charcode.

**FPDFFontIsCharEmbedded**

Checks whether a character is embedded.

**FPDFFontIsEmbedded**

Checks whether the font is an embedded font.

**FPDFFontIsStandardFont**

Checks whether the font is a standard font.

**FPDFFontIsUnicodeCompatible**

Checks whether the font is Unicode compatible.

**FPDFFontIsVertWriting**

Checks whether the font is vertical writing.

**FPDFFontLoadGlyphPath**

Gets a character's path data. for T1, TT, and CIDFont only.



**FPDFFontNew**

Creates a new PDF font object from a font dictionary.

**FPDFFontRecognizeChar**

Use OCR to recognize a character and return Unicode.

**FPDFFontUnicodeFromCharCode**

Gets a unicode string for a charcode.

**FPDFFontUnicodeFromCharCodeEx**

Gets a unicode string for a charcode. We will use "/Encoding" first if "/Encoding" is neither "Identity-H" nor "Identity-V".

**Functions detail****FPDFFontAppendChar****Syntax**

```
FS_INT32 FPDFFontAppendChar (
    FPD_Font font,
    FS_LPSTR inOutStr,
    FS_DWORD charcode
);
```

**Description**

Append a charcode to a string buffer.

**Parameter**

font	[In] The input PDF font object.
inOutStr	[In/Out] Input a string buffer and append a charcode to it.
charcode	[In] The charcode to append.

**Return**

The number of bytes appended to the string buffer.

**Head file reference**

fpd\_resourceTempl.h: 216

**FPDFFontAppendChar2****Syntax**

```
void FPDFFontAppendChar2 (
    FPD_Font font,
    FS_ByteString str,
    FS_DWORD charcode
);
```

**Description**

Append a charcode to a string buffer.

**Parameter**

font	[In] The input PDF font object.
str	[In/Out] Input a string buffer and append a charcode to it.
charcode	[In] The charcode to append.

**Return**

void.

**Head file reference**

fpd\_resourceTempl.h: 227

**FPDFFontCharCodeFromUnicode****Syntax**

```
FS_DWORD FPDFFontCharCodeFromUnicode (
    FPD\_Font font,
    FS\_WCHAR unicode
);
```

**Description**

Gets a charcode from a Unicode.

**Parameter**

font	[In] The input PDF font object.
unicode	[In] The charcode.

**Return**

The charcode for the unicode.

**Head file reference**

fpd\_resourceTempl.h: 269

**FPDFFontCountChar****Syntax**

```
FS_INT32 FPDFFontCountChar (
    FPD\_Font font,
    FS\_LPCSTR str,
```



```
FS_INT32 size  
);
```

**Description**

Gets the count of characters in a string.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

str	[In] The string buffer.
-----	-------------------------

---

size	[In] The length in bytes of the string.
------	---

---

**Return**

The count of characters in the string.

**Head file reference**

fpd\_resourceTempl.h: 205

**FPDFFontDecodeString****Syntax**

```
void FPDFFontDecodeString (  
    FPD_Font font,  
    FS_LPCSTR src,  
    FS_WideString* outResult  
) ;
```

**Description**

Decode a font string to unicode string.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

src	[In] The source font string.
-----	------------------------------

---

outResult	[Out] The decoded unicode string.
-----------	-----------------------------------

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 309

## FPDFFontDestroy

### Syntax

```
void FPDFFontDestroy (
    FPD\_Font font
);
```

### Description

Destroys the PDF font object.

### Parameter

---

font	[In] The input PDF font object.
------	---------------------------------

---

### Return

void

### Head file reference

fpd\_resourceTempl.h: 31

## FPDFFontEncodeString

### Syntax

```
void FPDFFontEncodeString (
    FPD\_Font font,
    FS\_LPCWSTR wszSrc,
    FS\_BytString* outEncodeString
);
```

### Description

Encode an unicode string to font string.

### Parameter

---

font	[In] The input PDF font object.
------	---------------------------------

---

wszSrc	[In] The unicode string.
--------	--------------------------

---

outEncodeString	[Out] The encoded font string.
-----------------	--------------------------------

---

### Return

void



**Head file reference**

fpd\_resourceTempl.h: 298

**FPDFFontFXFontGetFaceName****Syntax**

```
void FPDFFontFXFontGetFaceName (
    FPD_FXFont font,
    FS_ByteString* outFaceName
);
```

**Description**

Gets the face name of the Foxit GE font.

**Parameter**

---

font	[In] The input Foxit GE font object.
------	--------------------------------------

---

outFaceName	[Out] It receives the face name.
-------------	----------------------------------

---

**Return**

void.

**Head file reference**

fpd\_resourceTempl.h: 457

**Since**[SDK LATEEST VERSION > 2.1.0.4](#)**FPDFFontFXFontGetFamilyName****Syntax**

```
void FPDFFontFXFontGetFamilyName (
    FPD_FXFont font,
    FS_ByteString* outFamilyName
);
```

**Description**

Gets the family name of the Foxit GE font.

**Parameter**

---

font	[In] The input Foxit GE font object.
------	--------------------------------------

---

outFamilyName	[Out] It receives the face name.
---------------	----------------------------------

---



**Return**

void.

**Head file reference**

fpd\_resourceTempl.h: 479

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FPDFFontFXFontGetPsName****Syntax**

```
void FPDFFontFXFontGetPsName (
    FPD\_FXFont font,
    FS\_WideString* outPsName
);
```

**Description**

Gets the postscript name of the Foxit GE font.

**Parameter**

---

font	[In] The input Foxit GE font object.
------	--------------------------------------

---

outPsName	[Out] It receives the postscript name.
-----------	--

---

**Return**

void.

**Head file reference**

fpd\_resourceTempl.h: 468

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FPDFFontFXFontIsBold****Syntax**

```
FS_BOOL FPDFFontFXFontIsBold (
    FPD\_FXFont font
);
```

**Description**

Checks whether it is bold font or not.

**Parameter**

---

font	[In] The input Foxit GE font object.
------	--------------------------------------

---

**Return**

TRUE if it is bold font, otherwise not.

**Head file reference**

fpd\_resourceTempl.h: 500

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FPDFFontFXFontIsItalic****Syntax**

```
FS_BOOL FPDFFontFXFontIsItalic (
    FPD\_FXFont font
);
```

**Description**

Checks whether it is italic font or not.

**Parameter**

---

font	[In] The input Foxit GE font object.
------	--------------------------------------

---

**Return**

TRUE if it is italic font, otherwise not.

**Head file reference**

fpd\_resourceTempl.h: 490

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FPDFFontFXFontLoadSubst****Syntax**

```
FS_BOOL FPDFFontFXFontLoadSubst (
    FPD\_FXFont font,
    FS_LPCSTR lpsFaceName,
    FS_BOOL bTrueType,
    FS_DWORD flags,
    FS_INT32 weight,
    FS_INT32 nItalicAngle,
    FS_INT32 nCharsetCP,
    FS_BOOL bVertical
);
```



**Description**

Loads a font through font mapper.

**Parameter**

font	[In] The input Foxit GE font object.
lpsFaceName	[In] PostScript name.
bTrueType	[In] TrueType or Type1.
flags	[In] PDF font flags (see PDF Reference section 5.7.1).
weight	[In] Original font weight. 0 for not specified.
nItalicAngle	[In] Original font italic angle. 0 for not specified.
nCharsetCP	[In] Code page for charset (see Win32 GetACP()).
bVertical	[In] Whether vertical writing-mode. Sets it FALSE as default.

**Return**

TRUE for success, otherwise not.

**Head file reference**

fpd\_resourceTempl.h: 532

**Since**

[SDK\\_LATEEST\\_VERSION > 7.3.0.0](#)

**FPDFFontGetBaseFont****Syntax**

```
void FPDFFontGetBaseFont (
    FPD\_Font font,
    FS\_ByteString* outFontName
);
```

**Description**

Gets the base font name.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

outFontName	[Out] It receives the base font name.
-------------	---------------------------------------

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 76

**FPDFFontGetCharBBox****Syntax**

```
FS_Rect FPDFFontGetCharBBox (
    FPD\_Font font,
    Input
);
```

**Description**

Gets the bounding box of a character.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

Input	[In] The charcode a character code.
-------	-------------------------------------

---

**Return**

The bounding box of a character.

**Head file reference**

fpd\_resourceTempl.h: 397

**FPDFFontGetCharMap****Syntax**

```
FS_CharMap FPDFFontGetCharMap (
    FPD\_Font font
);
```

**Description**

Gets the character map.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

The character map.

**Head file reference**

fpd\_resourceTempl.h: 289

**FPDFFontGetCharSize****Syntax**

```
FS_INT32 FPDFFontGetCharSize (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

**Description**

Gets the number of bytes for the char code.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

---

charcode	[In] Input a charcode.
----------	------------------------

---

**Return**

The number of bytes for the char code.

**Head file reference**

fpd\_resourceTempl.h: 238

**FPDFFontGetCharTypeWidth****Syntax**

```
FS_INT32 FPDFFontGetCharTypeWidth (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

**Description**

Gets a character's real width in the font file.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---



---

charcode	[In] The character code.
----------	--------------------------

---

**Return**

The real width of the character

**Head file reference**

fpd\_resourceTempl.h: 387

**FPDFFontGetCharWidthF****Syntax**

```
FS_INT32 FPDFFontGetCharWidthF (
    FPD_Font font,
    FS_DWORD charcode
);
```

**Description**

Gets a character's PDF width.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

---

charcode	[In] The character code.
----------	--------------------------

---

**Return**

The PDF width of the character.

**Head file reference**

fpd\_resourceTempl.h: 377

**FPDFFontGetCIDFont****Syntax**

```
FPD_Font FPDFFontGetCIDFont (
    FPD_Font font
);
```

**Description**

Gets a CID font.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

A CID font.

**Head file reference**

fpd\_resourceTempl.h: 122

## FPDFFontGetFace

**Syntax**

```
FPD_FT_Face FPDFFontGetFace (
    FPD\_Font font
);
```

**Description**

Gets embedded or substituted FT font face.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

The embedded or substituted FT font face.

**Head file reference**

fpd\_resourceTempl.h: 185

## FPDFFontGetFlags

**Syntax**

```
FS_DWORD FPDFFontGetFlags (
    FPD\_Font font
);
```

**Description**

Gets the font flags.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

The font flags.

**Head file reference**

fpd\_resourceTempl.h: 95

## FPDFFontGetFontBBox



**Syntax**

```
FS_Rect FPDFFontGetFontBBox (
    FPD\_Font font
);
```

**Description**

Gets the font's bounding box.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

The font's bounding box.

**Head file reference**

fpd\_resourceTempl.h: 330

**FPDFFontGetFontDict****Syntax**

```
FPD_Object FPDFFontGetFontDict (
    FPD\_Font font
);
```

**Description**

Gets the font dictionary.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

The font dictionary.

**Head file reference**

fpd\_resourceTempl.h: 167

**FPDFFontGetFontFile****Syntax**

```
FPD_StreamAcc FPDFFontGetFontFile (
    FPD\_Font font
);
```

**Description**

Gets the font stream.

#### Parameter

---

font	[In] The input PDF font object.
------	---------------------------------

---

#### Return

The font stream.

#### Head file reference

fpd\_resourceTempl.h: 158

## FPDFFontGetFontType

#### Syntax

```
FS_INT32 FPDFFontGetFontType (
    FPD\_Font font
);
```

#### Description

Gets the font type.

#### Parameter

---

font	[In] The input PDF font object.
------	---------------------------------

---

#### Return

The font type.

#### Head file reference

fpd\_resourceTempl.h: 57

## FPDFFontGetFontTypeName

#### Syntax

```
void FPDFFontGetFontTypeName (
    FPD\_Font font,
    FS\_BytString* outName
);
```

#### Description

Gets the font type name.

#### Parameter

---

font	[In] The input PDF font object.
------	---------------------------------

---



---

outName	[Out] The font type name.
---------	---------------------------

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 66

**FPDFFontGetFXFont****Syntax**

```
FPD_FXFont FPDFFontGetFXFont (
    FPD\_Font font
);
```

**Description**

Gets the Foxit GE font.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

The Foxit GE font.

**Head file reference**

fpd\_resourceTempl.h: 447

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FPDFFontGetItalicAngle****Syntax**

```
FS_INT32 FPDFFontGetItalicAngle (
    FPD\_Font font
);
```

**Description**

Gets the angle, expressed in degrees counterclockwise from the vertical, of the dominant vertical strokes of the font.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---



**Return**

The angle.

**Head file reference**

fpd\_resourceTempl.h: 357

**FPDFFontGetNextChar****Syntax**

```
FS_DWORD FPDFFontGetNextChar (
    FPD\_Font font,
    FS\_LPCSTR str,
    FS\_INT32* inOutOffset
);
```

**Description**

Gets a charcode from a string at specified position.

**Parameter**

font	[In] The input PDF font object.
str	[In] The char buffer.
inOutOffset	[In/Out] Input the zero-based position and receive the next charcode position.

**Return**

A charcode with next char.

**Head file reference**

fpd\_resourceTempl.h: 194

**FPDFFontGetPDFDoc****Syntax**

```
FPD_Document FPDFFontGetPDFDoc (
    FPD\_Font font
);
```

**Description**

Gets the PDF document.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

The PDF document.

**Head file reference**

fpd\_resourceTempl.h: 407

**FPDFFontGetStockFont****Syntax**

```
FPD_Font FPDFFontGetStockFont (
    FPD\_Document fpdDoc,
    FS\_LPCSTR szFontName
);
```

**Description**

Gets the "stocked" fonts (Adobe base 14 fonts), which are always available for rendering activities. However, these fonts can't be directly referred in any document, unless the font resource is added to the document.

**Parameter**


---

fpdDoc	[In] The document associated with the font name.
--------	--

---

szFontName	[In] The font name.
------------	---------------------

---

**Return**

Return a pdf font.

**Head file reference**

fpd\_resourceTempl.h: 40

**Note:** List of currently supported standard fonts:

- Courier, Courier-Bold, Courier-BoldOblique, Courier-Oblique.
- Helvetica, Helvetica-Bold, Helvetica-BoldOblique, Helvetica-Oblique.
- Times-Roman, Times-Bold, Times-Italic, Times-BoldItalic.
- Symbol, ZapfDingbats.

**FPDFFontGetStringWidth****Syntax**

```
FS_INT32 FPDFFontGetStringWidth (
    FPD\_Font font,
    const FS\_CHAR* pCharBuf,
    FS\_INT32 size
```



);

**Description**

Gets a string's width.

**Parameter**

font	[In] The input PDF font object.
pCharBuf	[In] The char buffer. Must be in font encoding.
size	[In] The length in bytes of the string.

**Return**

The width of the string.

**Head file reference**

fpd\_resourceTempl.h: 366

**FPDFFontGetSubstFont****Syntax**

```
FPD_SubstFont FPDFFontGetSubstFont (
    FPD\_Font font
);
```

**Description**

Gets the substitute font.

**Parameter**

font	[In] The input PDF font object.
------	---------------------------------

**Return**

The substitute font.

**Head file reference**

fpd\_resourceTempl.h: 86

**FPDFFontGetSubstFontCharset****Syntax**

```
FS_INT32 FPDFFontGetSubstFontCharset (
    FPD\_SubstFont substFont
);
```



**Description**

Gets the charset of the substitution font.

**Parameter**

---

substFont	[In] The input substitution font object.
-----------	--

---

**Return**

The charset of the substitution font.

**Head file reference**

fpd\_resourceTempl.h: 427

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDFFontGetSubstFontWeight****Syntax**

```
FS_INT32 FPDFFontGetSubstFontWeight (
    FPD_SubstFont substFont
);
```

**Description**

Gets the weight parameter, non-zero for synthetic weight generation only.

**Parameter**

---

substFont	[In] The input substitution font object.
-----------	--

---

**Return**

The weight parameter, non-zero for synthetic weight generation only.

**Head file reference**

fpd\_resourceTempl.h: 522

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FPDFFontGetTrueTypeFont****Syntax**

```
FPD_Font FPDFFontGetTrueTypeFont (
    FPD_Font font
);
```



**Description**

Gets a true type font.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

A True-type font.

**Head file reference**

fpd\_resourceTempl.h: 113

**FPDFFontGetType1Font****Syntax**

```
FPD_Font FPDFFontGetType1Font (
    FPD\_Font font
);
```

**Description**

Gets a type1 font.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

A type1 font.

**Head file reference**

fpd\_resourceTempl.h: 104

**FPDFFontGetType3Font****Syntax**

```
FPD_Font FPDFFontGetType3Font (
    FPD\_Font font
);
```

**Description**

Gets a type3 font.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

A type3 font.

**Head file reference**

fpd\_resourceTempl.h: 131

**FPDFFontGetTypeAscent****Syntax**

```
FS_INT32 FPDFFontGetTypeAscent (
    FPD\_Font font
);
```

**Description**

Gets the typographic ascent.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

The typographic ascent.

**Head file reference**

fpd\_resourceTempl.h: 339

**FPDFFontGetTypeDescent****Syntax**

```
FS_INT32 FPDFFontGetTypeDescent (
    FPD\_Font font
);
```

**Description**

Gets the typographic descent, most negative.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

The typographic descent, most negative.

**Head file reference**

fpd\_resourceTempl.h: 348



## FPDFFontGlyphFromCharCode

### Syntax

```
FS_INT32 FPDFFontGlyphFromCharCode (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

### Description

Gets the glyph index for a charcode.

### Parameter

---

font	[In] The input PDF font object.
------	---------------------------------

---

charcode	[In] The charcode.
----------	--------------------

---

### Return

The glyph index of the charcode. Return -1 for unknown.

### Head file reference

fpd\_resourceTempl.h: 248

**Note:** For embedded font only.

## FPDFFontIsCharEmbedded

### Syntax

```
FS_BOOL FPDFFontIsCharEmbedded (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

### Description

Checks whether a character is embedded.

### Parameter

---

font	[In] The input PDF font object.
------	---------------------------------

---

charcode	[In] The charcode.
----------	--------------------

---

### Return

[TRUE](#) if a character is embedded, otherwise [FALSE](#).



**Head file reference**

fpd\_resourceTempl.h: 279

**FPDFFontIsEmbedded****Syntax**

```
FS_BOOL FPDFFontIsEmbedded (
    FPD_Font font
);
```

**Description**

Checks whether the font is an embedded font.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

[TRUE](#) if the font is an embedded font, otherwise [FALSE](#) .

**Head file reference**

fpd\_resourceTempl.h: 140

**FPDFFontIsStandardFont****Syntax**

```
FS_BOOL FPDFFontIsStandardFont (
    FPD_Font font
);
```

**Description**

Checks whether the font is a standard font.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

[TRUE](#) if the font is a standard font, otherwise [FALSE](#) .

**Head file reference**

fpd\_resourceTempl.h: 176

**FPDFFontIsUnicodeCompatible****Syntax**

```
FS_BOOL FPDFFontIsUnicodeCompatible (
    FPD\_Font font
);
```

**Description**

Checks whether the font is Unicode compatible.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

[TRUE](#) if the font is Unicode compatible, otherwise [FALSE](#).

**Head file reference**

fpd\_resourceTempl.h: 149

**FPDFFontIsVertWriting****Syntax**

```
FS_BOOL FPDFFontIsVertWriting (
    FPD\_Font font
);
```

**Description**

Checks whether the font is vertical writing.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

**Return**

[TRUE](#) if the PDF font is vertical writing.

**Head file reference**

fpd\_resourceTempl.h: 437

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FPDFFontLoadGlyphPath****Syntax**

```
FPD_Path FPDFFontLoadGlyphPath (
    FPD\_Font font,
    FS\_DWORD charcode,
    FS\_INT32 destWidth
```



);

**Description**

Gets a character's path data. for T1, TT, and CIDFont only.

**Parameter**

font	[In] The input PDF font object.
charcode	[In] The character code.
destWidth	[In] The destination's width.

**Return**

A character's path data. for T1, TT, and CIDFont only.

**Head file reference**

fpd\_resourceTempl.h: 416

**FPDFFontNew****Syntax**

```
FPD_Font FPDFFontNew (
    FPD\_Document doc,
    FPD\_Object dic
);
```

**Description**

Creates a new PDF font object from a font dictionary.

**Parameter**

doc	[In] The PDF document.
dic	[In] The font dictionary.

**Return**

A new PDF font object.

**Head file reference**

fpd\_resourceTempl.h: 21

**FPDFFontRecognizeChar**

**Syntax**

```
FS_WCHAR FPDFFontRecognizeChar (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

**Description**

Use OCR to recognize a character and return Unicode.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

charcode	[In] The charcode.
----------	--------------------

---

**Return**

The recognized unicode.

**Head file reference**

fpd\_resourceTempl.h: 320

**Note:** This function may be called when UnicodeFromCharCode() fails to get a character

**FPDFFontUnicodeFromCharCode****Syntax**

```
void FPDFFontUnicodeFromCharCode (
    FPD\_Font font,
    FS\_DWORD charcode,
    FS\_WideString* outUnicode
);
```

**Description**

Gets a unicode string for a charcode.

**Parameter**

---

font	[In] The input PDF font object.
------	---------------------------------

---

charcode	[In] The charcode.
----------	--------------------

---

outUnicode	[Out] The unicode string for the charcode. NULL for unknown.
------------	--

---

**Return****Head file reference**

---

fpd\_resourceTempl.h: 258

### **FPDFFontUnicodeFromCharCodeEx**

#### **Syntax**

```
void FPDFFontUnicodeFromCharCodeEx (
    FPD\_Font font,
    FS\_DWORD charcode,
    FS\_WideString* outUnicode
);
```

#### **Description**

Gets a unicode string for a charcode. We will use "/Encoding" first if "/Encoding" is neither "Identity-H" nor "Identity-V".

#### **Parameter**

font	[In] The input PDF font object.
charcode	[In] The charcode.
outUnicode	[Out] The unicode string for the charcode. NULL for unknown.

#### **Return**

void.

#### **Head file reference**

fpd\_resourceTempl.h: 510

#### **Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

## **FPD\_FontEncoding**

### [Return from Used by](#)

#### **Description**

Single byte font encoding used by T1 or TT fonts. See [FPDFFontEncodingNew](#) , [FPDFFontEncodingDestroy](#) .

#### **Returned from**

[FPDTrueTypeFontGetEncoding](#)  
[FPDTType1FontGetEncoding](#)  
[FPDTType3FontGetEncoding](#)



[\*\*FPDFFontEncodingNew\*\*](#)  
[\*\*FPDFFontEncodingNew2\*\*](#)

### Used by

[\*\*FPDDocAddStandardFont\*\*](#)  
[\*\*FPDFFontEncodingCharCodeFromUnicode\*\*](#)  
[\*\*FPDFFontEncodingDestroy\*\*](#)  
[\*\*FPDFFontEncodingIsIdentical\*\*](#)  
[\*\*FPDFFontEncodingLoadEncoding\*\*](#)  
[\*\*FPDFFontEncodingRealize\*\*](#)  
[\*\*FPDFFontEncodingSetUnicode\*\*](#)  
[\*\*FPDFFontEncodingUnicodeFromCharCode\*\*](#)

## Functions

### Functions summary

#### [\*\*FPDFFontEncodingCharCodeFromUnicode\*\*](#)

Gets the charcode from a unicode.

#### [\*\*FPDFFontEncodingDestroy\*\*](#)

Destroys the font encoding.

#### [\*\*FPDFFontEncodingIsIdentical\*\*](#)

Checks whether the encoding is identical with another encoding.

#### [\*\*FPDFFontEncodingLoadEncoding\*\*](#)

Loads encoding from a PDF encoding object.

#### [\*\*FPDFFontEncodingNew\*\*](#)

Creates a new single byte font encoding used by T1 or TT fonts.

#### [\*\*FPDFFontEncodingNew2\*\*](#)

Creates a new font encoding with a predefined encoding.

#### [\*\*FPDFFontEncodingRealize\*\*](#)

Realizes the font encoding in a PDF encoding object. Can be NULL if not needed.

#### [\*\*FPDFFontEncodingSetUnicode\*\*](#)

Sets the unicode of a charcode.

#### [\*\*FPDFFontEncodingUnicodeFromCharCode\*\*](#)

Gets the unicode from a charcode.

### Functions detail

#### [\*\*FPDFFontEncodingCharCodeFromUnicode\*\*](#)

##### **Syntax**

```
FS_INT32 FPDFFontEncodingCharCodeFromUnicode (
    FPD\_FontEncoding fontEncoding,
    FS\_WCHAR unicode
);
```

##### **Description**

Gets the charcode from a unicode.

##### **Parameter**



---

fontEncoding	[In] The input font encoding.
--------------	-------------------------------

---

unicode	[In] The unicode char code.
---------	-----------------------------

---

**Return**

The charcode of the unicode. -1 for not found.

**Head file reference**

fpd\_resourceTempl.h: 614

**FPDFFontEncodingDestroy****Syntax**

```
void FPDFFontEncodingDestroy (
    FPD\_FontEncoding fontEncoding
);
```

**Description**

Destroys the font encoding.

**Parameter**

---

fontEncoding	[In] The input font encoding.
--------------	-------------------------------

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 575

**FPDFFontEncodingIsIdentical****Syntax**

```
FS_BOOL FPDFFontEncodingIsIdentical (
    FPD\_FontEncoding fontEncoding,
    FPD\_FontEncoding another
);
```

**Description**

Checks whether the encoding is identical with another encoding.

**Parameter**

---

fontEncoding	[In] The input font encoding.
--------------	-------------------------------

---



---

another	[In] The another encoding.
---------	----------------------------

---

**Return**

TRUE means identical, FALSE not identical.

**Head file reference**

fpd\_resourceTempl.h: 594

## FPDFFontEncodingLoadEncoding

**Syntax**

```
void FPDFFontEncodingLoadEncoding (  
    FPD_FontEncoding fontEncoding,  
    FPD_Object encoding  
) ;
```

**Description**

Loads encoding from a PDF encoding object.

**Parameter**

---

fontEncoding	[In] The input font encoding.
--------------	-------------------------------

---

---

encoding	[In] The PDF object of encoding.
----------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 584

## FPDFFontEncodingNew

**Syntax**

```
FPD_FontEncoding FPDFFontEncodingNew (void);
```

**Description**

Creates a new single byte font encoding used by T1 or TT fonts.

**Return**

A new font encoding.

**Head file reference**

fpd\_resourceTempl.h: 557



## FPDFFontEncodingNew2

### Syntax

```
FPD_FontEncoding FPDFFontEncodingNew2 (
    int PredefinedEncoding
);
```

### Description

Creates a new font encoding with a predefined encoding.

### Parameter

---

PredefinedEncoding	[In] The input predefined encoding.
--------------------	-------------------------------------

---

### Return

A new font encoding.

### Head file reference

fpd\_resourceTempl.h: 566

## FPDFFontEncodingRealize

### Syntax

```
FPD_Object FPDFFontEncodingRealize (
    FPD\_FontEncoding fontEncoding
);
```

### Description

Realizes the font encoding in a PDF encoding object. Can be NULL if not needed.

### Parameter

---

fontEncoding	[In] The input font encoding.
--------------	-------------------------------

---

### Return

A new PDF encoding object corresponding to the encoding table.

### Head file reference

fpd\_resourceTempl.h: 635

## FPDFFontEncodingSetUnicode

### Syntax

```
void FPDFFontEncodingSetUnicode (
    FPD\_FontEncoding fontEncoding,
    FS\_BYTE charcode,
    FS\_WCHAR unicode
);
```



**Description**

Sets the unicode of a charcode.

**Parameter**

fontEncoding	[In] The input font encoding.
charcode	[In] Input a charcode.
unicode	[In] The new unicode value for the charcode.

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 624

**FPDFFontEncodingUnicodeFromCharCode****Syntax**

```
FS_WCHAR FPDFFontEncodingUnicodeFromCharCode (
    FPD\_FontEncoding fontEncoding,
    FS\_BYTE charcode
);
```

**Description**

Gets the unicode from a charcode.

**Parameter**

fontEncoding	[In] The input font encoding.
charcode	[In] The input charcode.

**Return**

The unicode of the charcode.

**Head file reference**

fpd\_resourceTempl.h: 604

# FPD\_Form



## [Return from Used by](#)

### Description

A [FPD\\_Form](#) is a self-contained set of graphics operators that is used when a particular graphic is drawn more than once in a document. It corresponds to a PDF XObject (see *Section 4.9, Form XObjects, the PDF Reference*). See [FPDFFormNew](#), [FPDFFormDestroy](#).

### Returned from

[FPDAAnnotGetAPForm](#)  
[FPDFFormControlGetField](#)  
[FPDFFormControlGetHighlightingMode](#)  
[FPDFFormControlGetType](#)  
[FPDFFormFieldGetControl](#)  
[FPDFFormFieldGetType](#)  
[FPDFFormNotifyNew](#)  
[FPDFFormObjectGetForm](#)  
[FPDInterFormAddControl](#)  
[FPDInterFormCheckRequiredFields](#)  
[FPDInterFormGetControl](#)  
[FPDInterFormGetControlAtPoint](#)  
[FPDInterFormGetControlByDict](#)  
[FPDInterFormGetField](#)  
[FPDInterFormGetFieldByDict](#)  
[FPDInterFormGetFieldInCalculationOrder](#)  
[FPDInterFormGetFormNotify](#)  
[FPDInterFormGetPageControl](#)  
[FPDInterFormNewControl](#)  
[FPDInterFormNewField](#)  
[FPDTilingPatternGetForm](#)  
[FPDType3CharGetForm](#)  
[FPDFFormClone](#)  
[FPDFFormNew](#)

### Used by

[FPDFFormControlDrawControl](#)  
[FPDFFormControlGetAction](#)  
[FPDFFormControlGetAdditionalAction](#)  
[FPDFFormControlGetBackgroundColor](#)  
[FPDFFormControlGetBorderColor](#)  
[FPDFFormControlGetCheckedAPState](#)  
[FPDFFormControlGetControlAlignment](#)  
[FPDFFormControlGetDefaultAppearance](#)  
[FPDFFormControlGetDefaultControlFont](#)  
[FPDFFormControlGetDownCaption](#)  
[FPDFFormControlGetDownIcon](#)  
[FPDFFormControlGetExportValue](#)  
[FPDFFormControlGetField](#)  
[FPDFFormControlGetHighlightingMode](#)  
[FPDFFormControlGetIconFit](#)  
[FPDFFormControlGetInterForm](#)  
[FPDFFormControlGetNormalCaption](#)



[FPDFFormControlGetNormalIcon](#)  
[FPDFFormControlGetOriginalBackgroundColor](#)  
[FPDFFormControlGetOriginalBackgroundColor2](#)  
[FPDFFormControlGetOriginalBorderColor](#)  
[FPDFFormControlGetOriginalBorderColor2](#)  
[FPDFFormControlGetRect](#)  
[FPDFFormControlGetRolloverCaption](#)  
[FPDFFormControlGetRolloverIcon](#)  
[FPDFFormControlGetRotation](#)  
[FPDFFormControlGetPosition](#)  
[FPDFFormControlGetType](#)  
[FPDFFormControlGetWidget](#)  
[FPDFFormControlHasMKEntry](#)  
[FPDFFormControlIsChecked](#)  
[FPDFFormControlIsDefaultChecked](#)  
[FPDFFormControlRemoveMKEntry](#)  
[FPDFFormControlSetAction](#)  
[FPDFFormControlSetAdditionalAction](#)  
[FPDFFormControlSetBackgroundColor](#)  
[FPDFFormControlSetBorderColor](#)  
[FPDFFormControlSetControlAlignment](#)  
[FPDFFormControlSetDefaultAppearance](#)  
[FPDFFormControlSetDefaultControlFont](#)  
[FPDFFormControlSetDownCaption](#)  
[FPDFFormControlSetDownCaptionW](#)  
[FPDFFormControlSetDownIcon](#)  
[FPDFFormControlSetExportValue](#)  
[FPDFFormControlSetHighlightingMode](#)  
[FPDFFormControlSetIconFit](#)  
[FPDFFormControlSetNormalCaption](#)  
[FPDFFormControlSetNormalCaptionW](#)  
[FPDFFormControlSetNormalIcon](#)  
[FPDFFormControlSetRolloverCaption](#)  
[FPDFFormControlSetRolloverCaptionW](#)  
[FPDFFormControlSetRolloverIcon](#)  
[FPDFFormControlSetRotation](#)  
[FPDFFormControlSetTextPosition](#)  
[FPDFFormFieldCheckControl](#)  
[FPDFFormFieldClearOptions](#)  
[FPDFFormFieldClearSelectedOptions](#)  
[FPDFFormFieldClearSelection](#)  
[FPDFFormFieldCountControls](#)  
[FPDFFormFieldCountOptions](#)  
[FPDFFormFieldCountSelectedItems](#)  
[FPDFFormFieldCountSelectedOptions](#)  
[FPDFFormFieldDefaultCheckControl](#)  
[FPDFFormFieldDeleteOption](#)  
[FPDFFormFieldFindOption](#)  
[FPDFFormFieldFindOptionValue](#)  
[FPDFFormFieldGetAdditionalAction](#)  
[FPDFFormFieldGetAlternateName](#)  
[FPDFFormFieldGetControl](#)  
[FPDFFormFieldGetControlIndex](#)  
[FPDFFormFieldGetDefaultSelectedItem](#)  
[FPDFFormFieldGetDefaultStyle](#)  
[FPDFFormFieldGetDefaultValue](#)  
[FPDFFormFieldGetFieldDict](#)

[FPDFFormFieldGetFieldFlags](#)  
[FPDFFormFieldGetFieldType](#)  
[FPDFFormFieldGetFlags](#)  
[FPDFFormFieldGetFont](#)  
[FPDFFormFieldGetFontSize](#)  
[FPDFFormFieldGetFullName](#)  
[FPDFFormFieldGetInterForm](#)  
[FPDFFormFieldGetMappingName](#)  
[FPDFFormFieldGetMaxLen](#)  
[FPDFFormFieldGetOptionLabel](#)  
[FPDFFormFieldGetOptionValue](#)  
[FPDFFormFieldGetRichTextString](#)  
[FPDFFormFieldGetSelectedIndex](#)  
[FPDFFormFieldGetSelectedOptionIndex](#)  
[FPDFFormFieldGetTopVisibleIndex](#)  
[FPDFFormFieldGetType](#)  
[FPDFFormFieldGetValue](#)  
[FPDFFormFieldInsertOption](#)  
[FPDFFormFieldIsItemDefaultSelected](#)  
[FPDFFormFieldIsItemSelected](#)  
[FPDFFormFieldIsOptionSelected](#)  
[FPDFFormFieldResetField](#)  
[FPDFFormFieldSelectOption](#)  
[FPDFFormFieldSetAdditionalAction](#)  
[FPDFFormFieldSetAlternateName](#)  
[FPDFFormFieldSetAlternateNameW](#)  
[FPDFFormFieldSetDefaultStyle](#)  
[FPDFFormFieldSetDefaultValue](#)  
[FPDFFormFieldSetFieldFlags](#)  
[FPDFFormFieldSetItemDefaultSelection](#)  
[FPDFFormFieldSetItemSelection](#)  
[FPDFFormFieldSetMappingName](#)  
[FPDFFormFieldSetMappingNameW](#)  
[FPDFFormFieldSetMaxLen](#)  
[FPDFFormFieldSetOptionLabel](#)  
[FPDFFormFieldSetOptionValue](#)  
[FPDFFormFieldSetRichTextString](#)  
[FPDFFormFieldSetRichTextStringW](#)  
[FPDFFormFieldSetTopVisibleIndex](#)  
[FPDFFormFieldSetValue](#)  
[FPDFFormFieldUpdateUnisonStatus](#)  
[FPDFFormNotifyDestroy](#)  
[FPDFFormNotifyNew](#)  
[FPDFFormObjectSetForm](#)  
[FPDInterFormDeleteControl](#)  
[FPDInterFormDeleteField2](#)  
[FPDInterFormRenameControl](#)  
[FPDInterFormRenameField2](#)  
[FPDRenderContextAppendForm](#)  
[FPDRenderContextDrawForm](#)  
[FPDFFormBackgroundAlphaNeeded](#)  
[FPDFFormCalcBoundingBox](#)  
[FPDFFormClone](#)  
[FPDFFormContinueParse](#)  
[FPDFFormCountObjects](#)  
[FPDFFormDebugOutput](#)  
[FPDFFormDestroy](#)

[\*\*FPDFFormEstimateParseProgress\*\*](#)  
[\*\*FPDFFormFindCSName\*\*](#)  
[\*\*FPDFFormFindFontName\*\*](#)  
[\*\*FPDFFormGenerateContent\*\*](#)  
[\*\*FPDFFormGetDict\*\*](#)  
[\*\*FPDFFormGetDocument\*\*](#)  
[\*\*FPDFFormGetFirstObjectPosition\*\*](#)  
[\*\*FPDFFormGetFormStream\*\*](#)  
[\*\*FPDFFormGetLastObjectPosition\*\*](#)  
[\*\*FPDFFormGetNextObject\*\*](#)  
[\*\*FPDFFormGetObjectAt\*\*](#)  
[\*\*FPDFFormGetObjectByIndex\*\*](#)  
[\*\*FPDFFormGetObjectIndex\*\*](#)  
[\*\*FPDFFormGetParseState\*\*](#)  
[\*\*FPDFFormGetPrevObject\*\*](#)  
[\*\*FPDFFormGetResourcesDictionary\*\*](#)  
[\*\*FPDFFormInsertObject\*\*](#)  
[\*\*FPDFFormIsParsed\*\*](#)  
[\*\*FPDFFormMoveObject\*\*](#)  
[\*\*FPDFFormNewContentGenerator\*\*](#)  
[\*\*FPDFFormRealizeResource\*\*](#)  
[\*\*FPDFFormRemoveObject\*\*](#)  
[\*\*FPDFFormReplaceObject\*\*](#)  
[\*\*FPDFFormSetResourcesDictionary\*\*](#)  
[\*\*FPDFFormStartParse\*\*](#)  
[\*\*FPDFFormTransform\*\*](#)

## Functions

### Functions summary

#### [\*\*FPDFFormBackgroundAlphaNeeded\*\*](#)

Checks whether this object list needs background alpha channel to render.

#### [\*\*FPDFFormCalcBoundingBox\*\*](#)

Calculates the bounding box of all form objects in the collection.

#### [\*\*FPDFFormClone\*\*](#)

Clones a form.

#### [\*\*FPDFFormContinueGenerateContent\*\*](#)

Continues to generate the PDF content progressively.

#### [\*\*FPDFFormContinueParse\*\*](#)

Continues parsing.

#### [\*\*FPDFFormCountObjects\*\*](#)

Gets the count of form objects in the collection.

#### [\*\*FPDFFormDebugOutput\*\*](#)

Outputs debug information. For debug only: list all form objects.

#### [\*\*FPDFFormDestroy\*\*](#)

Destroys the PDF form data object. If it is added into the page, it is taken over by page and don't destroy it.

#### [\*\*FPDFFormDestroyContentGenerator\*\*](#)

Destroys the PDF form content generator.

#### [\*\*FPDFFormEstimateParseProgress\*\*](#)

Estimates the percentage of parse progress.

#### [\*\*FPDFFormFindCSName\*\*](#)



Finds a resource name of specified color space.

**FPDFFormFindFontName**

Finds a resource name of specified font.

**FPDFFormGenerateContent**

Replaces the form content stream.

**FPDFFormGetDict**

Gets the form stream dictionary.

**FPDFFormGetDocument**

Gets the PDF document.

**FPDFFormGetFirstObjectPosition**

Gets the position of the first form object.

**FPDFFormGetFormStream**

Gets the form stream. For form only.

**FPDFFormGetLastObjectPosition**

Gets the position of the last form object

**FPDFFormGetNextObject**

Gets the current object and moves to next position.

**FPDFFormGetObjectAt**

Gets an object at specified position.

**FPDFFormGetObjectByIndex**

Gets an object by a zero-based form object index.

**FPDFFormGetObjectIndex**

Gets the zero-based form object index in the object array.

**FPDFFormGetParseState**

Gets the current parsing state.

**FPDFFormGetPrevObject**

Gets the current object and move to previous position

**FPDFFormGetResourcesDictionary**

Gets the PDF form resources dictionary.

**FPDFFormInsertObject**

Inserts a form object. To insert before all objects, use NULL for posInsertAfter.

**FPDFFormIsParsed**

Checks whether the content has been parsed into form objects.

**FPDFFormMoveObject**

Moves a form object from a position to another position.

**FPDFFormNew**

Creates a new empty PDF form object.

**FPDFFormNewContentGenerator**

Creates the PDF form content generator.

**FPDFFormParseContent**

Parses all contents.

**FPDFFormRealizeResource**

Adds a resource to current object list. Return the resource name.

**FPDFFormRemoveObject**

Removes a form object.

**FPDFFormReplaceObject**

Replaces a form object with a new form object.

**FPDFFormSetResourcesDictionary**

Sets the PDF form resources dictionary.

**FPDFFormStartGenerateContent**



Starts to generate the PDF content progressively.

**FPDFFormStartParse**

Start parsing

**FPDFFormTransform**

Transforms all objects.

## Functions detail

### FPDFFormBackgroundAlphaNeeded

**Syntax**

```
FS_BOOL FPDFFormBackgroundAlphaNeeded (
    FPD_Form form
);
```

**Description**

Checks whether this object list needs background alpha channel to render.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

**Return**

TRUE for this object list needing background alpha channel to render, otherwise not.

**Head file reference**

fpd\_pageTempl.h: 790

**Note:** If it's TRUE, then the application should better use ARGB device to render it, otherwise the objects may need more time to render. Please call this function after the form has been parsed.

### FPDFFormCalcBoundingBox

**Syntax**

```
FS_FloatRect FPDFFormCalcBoundingBox (
    FPD_Form form
);
```

**Description**

Calculates the bounding box of all form objects in the collection.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

**Return**

The bounding box of all form objects in the collection.



**Head file reference**

fpd\_pageTempl.h: 801

**FPDFFormClone****Syntax**

```
FPD_Form FPDFFormClone (
    FPD Form form
);
```

**Description**

Clones a form.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

**Return**A form cloned from *form* .**Head file reference**

fpd\_pageTempl.h: 902

**FPDFFormContinueGenerateContent****Syntax**

```
FPD_ProgressiveStatus FPDFFormContinueGenerateContent (
    FPD ContentGenerator generator,
    FS PauseHandler pause
);
```

**Description**

Continues to generate the PDF content progressively.

**Parameter**

---

generator	[In] The input PDF content generator.
-----------	---------------------------------------

---

pause	[In] The input pause handler. Creates the pause handler by <a href="#">FSPauseHandlerCreate</a> . Sets it NULL as default.
-------	--

---

**Return**

The status of generating PDF form content progressively.

**Head file reference**

fpd\_pageTempl.h: 948

**Since**

[SDK LATEST VERSION > 2.1.0.4](#)

## FPDFFormContinueParse

**Syntax**

```
void FPDFFormContinueParse (
    FPD\_Form form,
    FS\_PauseHandler pauseHandler
);
```

**Description**

Continues parsing.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

pauseHandler	[In] The user-supplied pause handler.
--------------	---------------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 610

## FPDFFormCountObjects

**Syntax**

```
FS_DWORD FPDFFormCountObjects (
    FPD\_Form form
);
```

**Description**

Gets the count of form objects in the collection.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

**Return**

The count of form objects in the collection.

**Head file reference**

fpd\_pageTempl.h: 698



## FPDFFormDebugOutput

### Syntax

```
void FPDFFormDebugOutput (
    FPD Form form,
    FS\_LPSTR szFileName
);
```

### Description

Outputs debug information. For debug only: list all form objects.

### Parameter

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

szFileName	[In] Input file name.
------------	-----------------------

---

### Return

void

### Head file reference

fpd\_pageTempl.h: 770

## FPDFFormDestroy

### Syntax

```
void FPDFFormDestroy (
    FPD Form form
);
```

### Description

Destroys the PDF form data object. If it is added into the page, it is taken over by page and don't destroy it.

### Parameter

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

### Return

void

### Head file reference

fpd\_pageTempl.h: 601



## FPDFFormDestroyContentGenerator

### Syntax

```
void FPDFFormDestroyContentGenerator (
```

[FPD\\_ContentGenerator](#) generator  
);

### Description

Destroys the PDF form content generator.

### Parameter

---

generator	[In] The input PDF form content generator.
-----------	--

---

### Return

void.

### Head file reference

fpd\_pageTempl.h: 926

### Since

[SDK LATEEST VERSION > 2.1.0.4](#)

## FPDFFormEstimateParseProgress

### Syntax

```
FS_INT32 FPDFFormEstimateParseProgress (
```

[FPD\\_Form](#) form  
);

### Description

Estimates the percentage of parse progress.

### Parameter

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

### Return

The percentage of parse progress.

### Head file reference

fpd\_pageTempl.h: 641

## FPDFFormFindCSName

### Syntax

```
void FPDFFormFindCSName (
```



```
FPD_Form form,  
FPD_ColorSpace cs,  
FS_ByteString* outResName  
);
```

**Description**

Finds a resource name of specified color space.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

cs	[In] The input color space.
----	-----------------------------

---

outResName	[Out] It receives the resource name of the color space.
------------	---

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 845

**FPDFFormFindFontName****Syntax**

```
void FPDFFormFindFontName (  
    FPD_Form form,  
    FPD_Font font,  
    FS_ByteString* outResName  
);
```

**Description**

Finds a resource name of specified font.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

font	[In] The input font.
------	----------------------

---

outResName	[Out] It receives the resource name of the font.
------------	--

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 856

## FPDFFormGenerateContent

### Syntax

```
void FPDFFormGenerateContent (
    FPD Form form
);
```

### Description

Replaces the form content stream.

### Parameter

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

### Return

void

### Head file reference

fpd\_pageTempl.h: 911

## FPDFFormGetDict

### Syntax

```
FPD_Object FPDFFormGetDict (
    FPD Form form
);
```

### Description

Gets the form stream dictionary.

### Parameter

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

### Return

The form stream dictionary.

### Head file reference

fpd\_pageTempl.h: 810

## FPDFFormGetDocument

### Syntax

```
FPD_Document FPDFFormGetDocument (
```



```
FPD_Form form  
);
```

**Description**

Gets the PDF document.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

**Return**

The PDF document.

**Head file reference**

fpd\_pageTempl.h: 819

**FPDFFormGetFirstObjectPosition****Syntax**

```
FS_POSITION FPDFFormGetFirstObjectPosition (  
    FPD_Form form  
) ;
```

**Description**

Gets the position of the first form object.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

**Return**

The position of the first form object.

**Head file reference**

fpd\_pageTempl.h: 650

**FPDFFormGetFormStream****Syntax**

```
FPD_Object FPDFFormGetFormStream (  
    FPD_Form form  
) ;
```

**Description**

Gets the form stream. For form only.



**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

**Return**

The form stream.

**Head file reference**

fpd\_pageTempl.h: 867

**FPDFFormGetLastObjectPosition****Syntax**

```
FS_POSITION FPDFFormGetLastObjectPosition (
    FPD\_Form form
);
```

**Description**

Gets the position of the last form object

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

**Return**

The position of the last form object

**Head file reference**

fpd\_pageTempl.h: 659

**FPDFFormGetNextObject****Syntax**

```
FPD_PageObject FPDFFormGetNextObject (
    FPD\_Form form,
    FS\_POSITION* inOutPos
);
```

**Description**

Gets the current object and moves to next position.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---



inOutPos	[In/Out] The input current position, and receives the next position.
----------	--

**Return**

A form object.

**Head file reference**

fpd\_pageTempl.h: 668

**FPDFormGetObjectAt****Syntax**

```
FPD_PageObject FPDFormGetObjectAt (
    FPD_Form form,
    FS_POSITION pos
);
```

**Description**

Gets an object at specified position.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

pos	[In] Specifies the position of the form object.
-----	---

---

**Return**

A form object.

**Head file reference**

fpd\_pageTempl.h: 688

**FPDFormGetObjectByIndex****Syntax**

```
FPD_PageObject FPDFormGetObjectByIndex (
    FPD_Form form,
    FS_INT32 index
);
```

**Description**

Gets an object by a zero-based form object index.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---



---

index	[In] Specifies the zero-based index of the form object.
-------	---

---

**Return**

A form object.

**Head file reference**

fpd\_pageTempl.h: 717

**FPDFFormGetObjectIndex****Syntax**

```
FS_INT32 FPDFFormGetObjectIndex (
    FPD\_Form form,
    FPD\_PageObject obj
);
```

**Description**

Gets the zero-based form object index in the object array.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

---

obj	[In] The form object pointer.
-----	-------------------------------

---

**Return**

The zero-based index of the form object.

**Head file reference**

fpd\_pageTempl.h: 707

**FPDFFormGetParseState****Syntax**

```
FS_INT32 FPDFFormGetParseState (
    FPD\_Form form
);
```

**Description**

Gets the current parsing state.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

**Return**

The current parsing state:

- FPD\_CONTENT\_NOT\_PARSED
- FPD\_CONTENT\_PARSING
- FPD\_CONTENT\_PARSED

#### **Head file reference**

fpd\_pageTempl.h: 620

### **FPDFormGetPrevObject**

#### **Syntax**

```
FPD_PageObject FPDFormGetPrevObject (
    FPD\_Form form,
    FS\_POSITION* inOutPos
);
```

#### **Description**

Gets the current object and move to previous position

#### **Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

inOutPos	[In/Out] It input the current position, and receives the previous position.
----------	---

---

#### **Return**

A form object.

#### **Head file reference**

fpd\_pageTempl.h: 678

### **FPDFormGetResourcesDictionary**

#### **Syntax**

```
FPD_Object FPDFormGetResourcesDictionary (
    FPD\_Form form
);
```

#### **Description**

Gets the PDF form resources dictionary.

#### **Parameter**



---

form	[In] The input PDF form data object.
------	--------------------------------------

---

**Return**

The PDF form resources dictionary.

**Head file reference**

fpd\_pageTempl.h: 964

**Related method****Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FPDFFormInsertObject****Syntax**

```
FS_POSITION FPDFFormInsertObject (
    FPD\_Form form,
    FS\_POSITION posInsertAfter,
    FPD\_PageObject newObj
);
```

**Description**

Inserts a form object. To insert before all objects, use NULL for posInsertAfter.

**Parameter**


---

form	[In] The input PDF form data object.
------	--------------------------------------

---



---

posInsertAfter	[In] Specifies the position to insert after.
----------------	--

---



---

newObj	[In] The input new form object.
--------	---------------------------------

---

**Return**

The position of inserted form object

**Head file reference**

fpd\_pageTempl.h: 738

**FPDFFormIsParsed****Syntax**

```
FS_BOOL FPDFFormIsParsed (
    FPD\_Form form
);
```



**Description**

Checks whether the content has been parsed into form objects.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

**Return**

[TRUE](#) if the content has been parsed into form objects, otherwise [FALSE](#) .

**Head file reference**

fpd\_pageTempl.h: 632

**FPDFormMoveObject****Syntax**

```
FS_POSITION FPDFormMoveObject (
    FPD Form form,
    FS POSITION pos,
    FS POSITION newPosAfter
);
```

**Description**

Moves a form object from a position to another position.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

---

pos	[In] The original position of the form object.
-----	--

---

---

newPosAfter	[In] The new position to move after.
-------------	--------------------------------------

---

**Return**

The new position of the form object.

**Head file reference**

fpd\_pageTempl.h: 759

**FPDFormNew****Syntax**

```
FPD_Form FPDFormNew (
    FPD Document doc,
    FPD Object pageResources,
    FPD Object formStream
);
```



**Description**

Creates a new empty PDF form object.

**Parameter**

doc	[In] The PDF document.
pageResources	[In] The page resources.
formStream	[In] The form stream.

**Return**

A new empty PDF form data object.

**Head file reference**

fpd\_pageTempl.h: 590

**FPDFFormNewContentGenerator****Syntax**

```
FPD_ContentGenerator FPDFFormNewContentGenerator (
    FPD Form form
);
```

**Description**

Creates the PDF form content generator.

**Parameter**

form	[In] The input PDF form data object.
------	--------------------------------------

**Return**

The PDF form content generator.

**Head file reference**

fpd\_pageTempl.h: 920

**Related method**

[FPDFFormDestroyContentGenerator](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)



**FPDFormParseContent****Syntax**

```
void FPDFormParseContent (
    form,
    FPD\_AllStates graphicStates,
    FS\_AffineMatrix* pParentMatrix,
    FPD\_Type3Char Type3Char,
    FPD\_ParseOptions opts
);
```

**Description**

Parses all contents.

**Parameter**

form	[In] The input PDF form data object.
graphicStates	[In] The current graphics states. Set to NULL for current version.
pParentMatrix	[In] Matrix from form object to parent page/form. Optional.
Type3Char	[In] Used only when parsing Type3 character forms.
opts	[In] Parsing options.

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 889

**FPDFormRealizeResource****Syntax**

```
void FPDFormRealizeResource (
    FPD\_Form form,
    FPD\_Object pageResObj,
    FS\_MapPtrToPtr objMapping,
    FS\_LPSTR szType,
    FPD\_Object* outResObj,
    FS\_ByteString* outResName
);
```

**Description**

Adds a resource to current object list. Return the resource name.



**Parameter**


---

form	[In] The input PDF form data object.
pageResObj	[In] The input resource object
objMapping	[In] The input object mapping from object number to object pointer.
szType	[In] The resource type name.
outResObj	[Out] It receives the result resource object.
outResName	[Out] It receives the resource name.

---

**Return**

The resource name.

**Head file reference**

fpd\_pageTempl.h: 828

**Note:** Will try to use an existing resource first, if available. Caller should not release the result resource object, which may be same as the input object. The input object can be an external object (which comes from another document, or archive), in this case, the object mapping should be specified.

**FPDFFormRemoveObject****Syntax**

```
void FPDFFormRemoveObject (
    FPD Form form,
    FS POSITION pos
);
```

**Description**

Removes a form object.

**Parameter**


---

form	[In] The input PDF form data object.
pos	[In] Specifies the position of the form object to be removed.

---



**Return**

void

**Head file reference**

fpd\_pageTempl.h: 749

**FPDFFormReplaceObject****Syntax**

```
void FPDFFormReplaceObject (
    FPD Form form,
    FS POSITION pos,
    FPD PageObject newObj
);
```

**Description**

Replaces a form object with a new form object.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

pos	[In] Specifies the position of the form object to be replaced.
-----	--

---

newObj	[In] The input new form object.
--------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 727

**FPDFFormSetResourcesDictionary****Syntax**

```
void FPDFFormSetResourcesDictionary (
    FPD Form form,
    FPD Object resourcesDict
);
```

**Description**

Sets the PDF form resources dictionary.

**Parameter**

---

form	[In] The input PDF form data object.
------	--------------------------------------

---

resourcesDict	[In] The input PDF form resources dictionary.
---------------	---

---

**Return**

void.

**Head file reference**

fpd\_pageTempl.h: 975

**Related method****Since**[SDK LATEEST VERSION > 7.2.2](#)**FPDFFormStartGenerateContent****Syntax**

```
FS_BOOL FPDFFormStartGenerateContent (
    FPD\_ContentGenerator generator,
    FS\_FileStream fileStream
);
```

**Description**

Starts to generate the PDF content progressively.

**Parameter**


---

generator	[In] The input PDF content generator.
-----------	---------------------------------------

---

fileStream	[In] The input file stream object. Sets it NULL as default.
------------	---

---

**Return**

TRUE for success, otherwise not.

**Head file reference**

fpd\_pageTempl.h: 941

**Related method**[FPDFFormContinueGenerateContent](#)**Since**[SDK LATEEST VERSION > 2.1.0.4](#)**FPDFFormStartParse**

**Syntax**

```
void FPDFormStartParse (
    FPD Form form,
    FPD AllStates graphicStates,
    FS\_AffineMatrix* pParentMatrix,
    FPD\_Type3Char Type3Char,
    FPD\_ParseOptions opts
);
```

**Description**

Start parsing

**Parameter**

form	[In] The input PDF form data object.
graphicStates	[In] The current graphics states. Set to NULL for current version.
pParentMatrix	[In] Matrix from form object to parent page/form. Optional.
Type3Char	[In] Used only when parsing Type3 character forms.
opts	[In] Parsing options.

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 876

**FPDFormTransform****Syntax**

```
void FPDFormTransform (
    FPD Form form,
    FS\_AffineMatrix matrix
);
```

**Description**

Transforms all objects.

**Parameter**

form	[In] The input PDF form data object.
------	--------------------------------------



matrix [In] The input transform matrix.

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 780

## FPD\_FormControl

**[Return from Used by](#)**

### Description

A FPD\_FormControl is an appearance controller, it used to draw an annotation's presentation on the page or set the appearance data to change an annotation's appearance. See [FPDInterFormNewControl](#) , [FPDInterFormGetControl](#) , [FPDInterFormGetPageControl](#) , [FPDInterFormGetControlAtPoint](#) , [FPDInterFormGetControlByDict](#) , [FPDInterFormDeleteControl](#) .

### Returned from

[FPDFFormFieldGetControl](#)  
[FPDInterFormAddControl](#)  
[FPDInterFormGetControl](#)  
[FPDInterFormGetControlAtPoint](#)  
[FPDInterFormGetControlByDict](#)  
[FPDInterFormGetPageControl](#)  
[FPDInterFormNewControl](#)

### Used by

[FPDFFormFieldGetControlIndex](#)  
[FPDInterFormDeleteControl](#)  
[FPDInterFormRenameControl](#)  
[FPDFFormControlDrawControl](#)  
[FPDFFormControlGetAction](#)  
[FPDFFormControlGetAdditionalAction](#)  
[FPDFFormControlGetBackgroundColor](#)  
[FPDFFormControlGetBorderColor](#)  
[FPDFFormControlGetCheckedAPState](#)  
[FPDFFormControlGetControlAlignment](#)  
[FPDFFormControlGetDefaultAppearance](#)  
[FPDFFormControlGetDefaultControlFont](#)  
[FPDFFormControlGetDownCaption](#)  
[FPDFFormControlGetDownIcon](#)  
[FPDFFormControlGetExportValue](#)  
[FPDFFormControlGetField](#)  
[FPDFFormControlGetHighlightingMode](#)  
[FPDFFormControlGetIconFit](#)  
[FPDFFormControlGetInterForm](#)  
[FPDFFormControlGetNormalCaption](#)

[FPDFFormControlGetNormalIcon](#)  
[FPDFFormControlGetOriginalBackgroundColor](#)  
[FPDFFormControlGetOriginalBackgroundColor2](#)  
[FPDFFormControlGetOriginalBorderColor](#)  
[FPDFFormControlGetOriginalBorderColor2](#)  
[FPDFFormControlGetRect](#)  
[FPDFFormControlGetRolloverCaption](#)  
[FPDFFormControlGetRolloverIcon](#)  
[FPDFFormControlGetRotation](#)  
[FPDFFormControlGetPosition](#)  
[FPDFFormControlGetType](#)  
[FPDFFormControlGetWidget](#)  
[FPDFFormControlHasMKEntry](#)  
[FPDFFormControlIsChecked](#)  
[FPDFFormControlIsDefaultChecked](#)  
[FPDFFormControlRemoveMKEntry](#)  
[FPDFFormControlSetAction](#)  
[FPDFFormControlSetAdditionalAction](#)  
[FPDFFormControlSetBackgroundColor](#)  
[FPDFFormControlSetBorderColor](#)  
[FPDFFormControlSetControlAlignment](#)  
[FPDFFormControlSetDefaultAppearance](#)  
[FPDFFormControlSetDefaultControlFont](#)  
[FPDFFormControlSetDownCaption](#)  
[FPDFFormControlSetDownCaptionW](#)  
[FPDFFormControlSetDownIcon](#)  
[FPDFFormControlSetExportValue](#)  
[FPDFFormControlSetHighlightingMode](#)  
[FPDFFormControlSetIconFit](#)  
[FPDFFormControlSetNormalCaption](#)  
[FPDFFormControlSetNormalCaptionW](#)  
[FPDFFormControlSetNormalIcon](#)  
[FPDFFormControlSetRolloverCaption](#)  
[FPDFFormControlSetRolloverCaptionW](#)  
[FPDFFormControlSetRolloverIcon](#)  
[FPDFFormControlSetRotation](#)  
[FPDFFormControlSetTextPosition](#)

## Definitions

### Definitions summary

#### [FPD\\_FORM\\_FIELDTYPE\\_CHECKBOX](#)

Check box.

#### [FPD\\_FORM\\_FIELDTYPE\\_COMBOBOX](#)

Combo box.

#### [FPD\\_FORM\\_FIELDTYPE\\_LISTBOX](#)

List box.

#### [FPD\\_FORM\\_FIELDTYPE\\_PUSHBUTTON](#)

Push button.

#### [FPD\\_FORM\\_FIELDTYPE\\_RADIOBUTTON](#)

Radio button.

#### [FPD\\_FORM\\_FIELDTYPE\\_SIGNATURE](#)

Signature.

**FPD\_FORM\_FIELDTYPE\_TEXTFIELD**

Text field.

**FPD\_FORM\_FIELDTYPE\_UNKNOWN**

Unknown.

**Definitions detail****FPD\_FORM\_FIELDTYPE\_CHECKBOX****Syntax**

```
#define FPD_FORM_FIELDTYPE_CHECKBOX 2
```

**Description**

Check box.

**Group**

[FPDFFieldTypes](#)

**Head file reference**

fpd\_docExpT.h: 801

**FPD\_FORM\_FIELDTYPE\_COMBOBOX****Syntax**

```
#define FPD_FORM_FIELDTYPE_COMBOBOX 4
```

**Description**

Combo box.

**Group**

[FPDFFieldTypes](#)

**Head file reference**

fpd\_docExpT.h: 805

**FPD\_FORM\_FIELDTYPE\_LISTBOX****Syntax**

```
#define FPD_FORM_FIELDTYPE_LISTBOX 5
```

**Description**

List box.

**Group**

[FPDFFieldTypes](#)

**Head file reference**

fpd\_docExpT.h: 807



## FPD\_FORM\_FIELDTYPE\_PUSHBUTTON

### Syntax

```
#define FPD_FORM_FIELDTYPE_PUSHBUTTON 1
```

### Description

Push button.

### Group

[FPDFFieldTypes](#)

### Head file reference

fpd\_docExpT.h: 799

## FPD\_FORM\_FIELDTYPE\_RADIOBUTTON

### Syntax

```
#define FPD_FORM_FIELDTYPE_RADIOBUTTON 3
```

### Description

Radio button.

### Group

[FPDFFieldTypes](#)

### Head file reference

fpd\_docExpT.h: 803

## FPD\_FORM\_FIELDTYPE\_SIGNATURE

### Syntax

```
#define FPD_FORM_FIELDTYPE_SIGNATURE 7
```

### Description

Signature.

### Group

[FPDFFieldTypes](#)

### Head file reference

fpd\_docExpT.h: 811

## FPD\_FORM\_FIELDTYPE\_TEXTFIELD

### Syntax



```
#define FPD_FORM_FIELDTYPE_TEXTFIELD 6
```

**Description**

Text field.

**Group**

[FPDFieldTypes](#)

**Head file reference**

fpd\_docExpT.h: 809

## FPD\_FORM\_FIELDTYPE\_UNKNOWN

**Syntax**

```
#define FPD_FORM_FIELDTYPE_UNKNOWN 0
```

**Description**

Unknown.

**Group**

[FPDFieldTypes](#)

**Head file reference**

fpd\_docExpT.h: 797

## Functions

### Functions summary

[\*\*FPDFFormControlDrawControl\*\*](#)

Draws the form control.

[\*\*FPDFFormControlGetAction\*\*](#)

Gets the action to be performed when the annotation is activated.

[\*\*FPDFFormControlGetAdditionalAction\*\*](#)

Gets the additional-actions defining the annotation's behavior in response to various trigger events.

[\*\*FPDFFormControlGetBackgroundColor\*\*](#)

Gets the background color.

[\*\*FPDFFormControlGetBorderColor\*\*](#)

Gets the border color.

[\*\*FPDFFormControlGetCheckedAPState\*\*](#)

Gets the checked state appearance string.

[\*\*FPDFFormControlGetControlAlignment\*\*](#)

Gets the alignment of the control.

[\*\*FPDFFormControlGetDefaultAppearance\*\*](#)

Gets the default appearance.

[\*\*FPDFFormControlGetDefaultControlFont\*\*](#)

Gets the default font of the control.



**FPDFControlGetDownCaption**

Gets the widget annotation's alternate (down) caption.

**FPDFControlGetDownIcon**

Gets the widget annotation's down icon.

**FPDFControlGetExportValue**

Gets the export mapping name.

**FPDFControlGetField**

Gets the field it belongs to.

**FPDFControlGetHighlightingMode**

Gets the highlighting mode.

**FPDFControlGetIconFit**

Gets the icon fit of the widget.

**FPDFControlGetInterForm**

Gets the interactive form it belongs to.

**FPDFControlGetNormalCaption**

Gets the widget annotation's normal caption.

**FPDFControlGetNormalIcon**

Gets the widget annotation's normal icon.

**FPDFControlGetOriginalBackgroundColor**

Gets an original color component of the background color.

**FPDFControlGetOriginalBackgroundColor2**

Gets the original color of the background.

**FPDFControlGetOriginalBorderColor**

Gets an original color component of the border color.

**FPDFControlGetOriginalBorderColor2**

Gets the original color of the border.

**FPDFControlGetRect**

Gets the rectangle of the widget.

**FPDFControlGetRolloverCaption**

Gets the widget annotation's rollover caption.

**FPDFControlGetRolloverIcon**

Gets the widget annotation's rollover icon.

**FPDFControlGetRotation**

Gets the number of degrees by which the widget annotation is rotated counterclockwise relative to the page. The value must be a multiple of 90.

**FPDFControlGetPosition**

Gets where to position the text of the widget annotation's caption relative to its icon.

**FPDFControlGetType**

Gets the field type.

**FPDFControlGetWidget**

Gets the widget annotation dictionary.

**FPDFControlHasMKEntry**

Checks whether the specified entry exist in the appearance characteristics dictionary.

**FPDFControlIsChecked**

Checks whether the control is checked.

**FPDFControlIsDefaultChecked**

Checks whether the control is default checked.

**FPDFControlRemoveMKEntry**

Removes a entry in the appearance characteristics dictionary.

**FPDFControlSetAction**

Sets the action to be performed when the annotation is activated.



**FPDFControlSetAdditionalAction**

Sets the additional-actions.

**FPDFControlSetBackgroundColor**

Sets the background color.

**FPDFControlSetBorderColor**

Sets the border color.

**FPDFControlSetControlAlignment**

Sets the alignment of the control.

**FPDFControlSetDefaultAppearance**

Sets the default appearance.

**FPDFControlSetDefaultControlFont**

Sets the default font of the control.

**FPDFControlSetDownCaption**

Sets the widget annotation's down caption. ANSI version.

**FPDFControlSetDownCaptionW**

Sets the widget annotation's down caption. Unicode version.

**FPDFControlSetDownIcon**

Sets the widget annotation's down icon.

**FPDFControlSetExportValue**

Sets the export mapping name.

**FPDFControlSetHighlightingMode**

Sets the highlighting mode.

**FPDFControlSetIconFit**

Sets the icon fit of the widget.

**FPDFControlSetNormalCaption**

Sets the widget annotation's normal caption. ANSI version.

**FPDFControlSetNormalCaptionW**

Sets the widget annotation's normal caption. Unicode version.

**FPDFControlSetNormalIcon**

Sets the widget annotation's normal icon.

**FPDFControlSetRolloverCaption**

Sets the widget annotation's rollover caption. ANSI version.

**FPDFControlSetRolloverCaptionW**

Sets the widget annotation's rollover caption. Unicode version.

**FPDFControlSetRolloverIcon**

Sets the widget annotation's rollover icon.

**FPDFControlSetRotation**

Sets the rotation.

**FPDFControlSetTextPosition**

Sets the text position.

## Functions detail

### FPDFControlDrawControl

#### Syntax

```
void FPDFControlDrawControl (  
    FPD_FormControl formContrl,  
    FPD_RenderDevice device,  
    FS_AffineMatrix matrix,  
    FPD_Page page,  
    FPD_AnnotAppearanceMode eMode
```



);

**Description**

Draws the form control.

**Parameter**

formContrl	[In] The input PDF interactive form control.
device	[In] The device to draw on.
matrix	[In] The transformation matrix from form control space to device space.
page	[In] The PDF page it belongs to.
eMode	[In] The input appearance mode.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5560

**FPDFFormControlGetAction****Syntax**

```
void FPDFFormControlGetAction (
    FPD\_FormControl formContrl,
    FPD\_Action* outAction
);
```

**Description**

Gets the action to be performed when the annotation is activated.

**Parameter**

formContrl	[In] The input PDF interactive form control.
outAction	[Out] The action to be performed when the annotation is activated.

**Return**

void



**Head file reference**

fpd\_docTempl.h: 5953

**FPDFFormControlGetAdditionalAction****Syntax**

```
void FPDFFormControlGetAdditionalAction (
    FPD FormControl formContrl,
    FPD AAction* pAAction
);
```

**Description**

Gets the additional-actions defining the annotation's behavior in response to various trigger events.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

pAAction	[Out] It receives the additional-actions.
----------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5973

**FPDFFormControlGetBackgroundColor****Syntax**

```
FS_ARGB FPDFFormControlGetBackgroundColor (
    FPD FormControl formContrl,
    FS INT32* outColorType
);
```

**Description**

Gets the background color.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

outColorType	[Out] It receives the color type.
--------------	-----------------------------------

---

**Return**

The FS\_ARGB color.c

**Head file reference**

fpd\_docTempl.h: 5723

## FPDFFormControlGetBorderColor

**Syntax**

```
FS_ARGB FPDFFormControlGetBorderColor (
    FPDFormControl formContrl,
    FS\_INT32* iColorType
);
```

**Description**

Gets the border color.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

iColorType	[Out] It receives the color type.
------------	-----------------------------------

---

**Return**

The FX\_ARGB color.

**Head file reference**

fpd\_docTempl.h: 5681

## FPDFFormControlGetCheckedAPState

**Syntax**

```
void FPDFFormControlGetCheckedAPState (
    FPDFormControl formContrl,
    FS\_ByteString* outState
);
```

**Description**

Gets the checked state appearance string.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

outState	[Out] The checked state appearance string.
----------	--

---



**Return**

void

**Head file reference**

fpd\_docTempl.h: 5573

**FPDFFormControlGetControlAlignment****Syntax**

```
FS_INT32 FPDFFormControlGetControlAlignment (
    FPD FormControl formContrl
);
```

**Description**

Gets the alignment of the control.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

**Return**

The alignment of the control.

- 0 - left alignment, the default setting.
- 1 - centered,
- 2 - right alignment.

**Head file reference**

fpd\_docTempl.h: 6033

**FPDFFormControlGetDefaultAppearance****Syntax**

```
void FPDFFormControlGetDefaultAppearance (
    FPD FormControl formContrl,
    FPD DefaultAppearance* outDefAP
);
```

**Description**

Gets the default appearance.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---



---

outDefAP	[Out] The default appearance.
----------	-------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5993

**Note:** Do not forget to save default appearance after you change any setting in it when you retrieve an object of FPD\_DefaultAppearance by GetDefaultAppearance method.

**FPDFormControlGetDefaultControlFont****Syntax**

```
FPD_Font FPDFormControlGetDefaultControlFont (
    FPD FormControl formContrl
);
```

**Description**

Gets the default font of the control.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

**Return**

The default font of the control.

**Head file reference**

fpd\_docTempl.h: 6014

**FPDFormControlGetDownCaption****Syntax**

```
void FPDFormControlGetDownCaption (
    FPD FormControl formContrl,
    FS WideString* outCaption
);
```

**Description**

Gets the widget annotation's alternate (down) caption.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---



---

outCaption	[Out] The widget annotation's alternate (down) caption.
------------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5825

**FPDFFormControlGetDownIcon****Syntax**

```
FPD_Object FPDFFormControlGetDownIcon (
    FPD FormControl formContrl
);
```

**Description**

Gets the widget annotation's down icon.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

**Return**

The widget annotation's down icon.

**Head file reference**

fpd\_docTempl.h: 5893

**FPDFFormControlGetExportValue****Syntax**

```
void FPDFFormControlGetExportValue (
    FPD FormControl formContrl,
    FS WideString* outValue
);
```

**Description**

Gets the export mapping name.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

---

outValue	[Out] The export mapping name.
----------	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5583

**FPDFFormControlGetField****Syntax**

```
FPD_FormField FPDFFormControlGetField (
    FPD FormControl formContrl
);
```

**Description**

Gets the field it belongs to.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

**Return**

The field it belongs to.

**Head file reference**

fpd\_docTempl.h: 5532

**FPDFFormControlGetHighlightingMode****Syntax**

```
FPD_FormCtrlHighlightingMode FPDFFormControlGetHighlightingMode (
    FPD FormControl formContrl
);
```

**Description**

Gets the highlighting mode.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

**Return**

The highlighting mode.

**Head file reference**

fpd\_docTempl.h: 5622

**FPDFFormControlGetIconFit**

**Syntax**

```
void FPDFormControlGetIconFit (
    FPD FormControl formContrl,
    FPD IconFit* outIconFit
);
```

**Description**

Gets the icon fit of the widget.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

outIconFit	[Out] The icon fit of the widget.
------------	-----------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5912

**FPDFormControlGetInterForm****Syntax**

```
FPD_InterForm FPDFormControlGetInterForm (
    FPD FormControl formContrl
);
```

**Description**

Gets the interactive form it belongs to.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

**Return**

The interactive form it belongs to.

**Head file reference**

fpd\_docTempl.h: 5523

**FPDFormControlGetNormalCaption****Syntax**

```
void FPDFormControlGetNormalCaption (
    FPD FormControl formContrl,
    FS WideString* outCaption
```



);

**Description**

Gets the widget annotation's normal caption.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

outCaption	[Out] The widget annotation's normal caption.
------------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5765

**FPDFFormControlGetNormalIcon****Syntax**

```
FPD_Object FPDFFormControlGetNormalIcon (
    FPD FormControl formContrl
);
```

**Description**

Gets the widget annotation's normal icon.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

**Return**

The widget annotation's normal icon.

**Head file reference**

fpd\_docTempl.h: 5855

**FPDFFormControlGetOriginalBackgroundColor****Syntax**

```
FS_FLOAT FPDFFormControlGetOriginalBackgroundColor (
    FPD FormControl formContrl,
    FS INT32 index
);
```



**Description**

Gets an original color component of the background color.

**Parameter**


---

formContrl	[In] The input PDF interactive form control.
index	[In] The component index of the original color.

---

**Return**

The value of the component.

**Head file reference**

fpd\_docTempl.h: 5733

**FPDFormControlGetOriginalBackgroundColor2****Syntax**

```
void FPDFormControlGetOriginalBackgroundColor2 (
    FPDFormControl formContrl,
    FS_INT32* outColorType,
    FS_FLOAT fc[4]
);
```

**Description**

Gets the original color of the background.

**Parameter**


---

formContrl	[In] The input PDF interactive form control.
outColorType	[Out] It receives the color type.
fc[4]	[Out] It receives the values of the components.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5743

**FPDFormControlGetOriginalBorderColor****Syntax**

```
FS_FLOAT FPDFormControlGetOriginalBorderColor (
    FPDFormControl formContrl,
```



```
FS_INT32 index  
);
```

**Description**

Gets an original color component of the border color.

**Parameter**

formContrl	[In] The input PDF interactive form control.
index	[In] The component index of the original color.

**Return**

The value of the component.

**Head file reference**

fpd\_docTempl.h: 5691

**FPDFFormControlGetOriginalBorderColor2****Syntax**

```
void FPDFFormControlGetOriginalBorderColor2 (  
    FPD_FormControl formContrl,  
    FS_INT32* outColorType,  
    FS_FLOAT outFc[4]  
) ;
```

**Description**

Gets the original color of the border.

**Parameter**

formContrl	[In] The input PDF interactive form control.
outColorType	[Out] It receives the color type.
outFc[4]	[Out] It receives the values of the components.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5701



## FPDFFormControlGetRect

### Syntax

```
void FPDFFormControlGetRect (
```

[FPD\\_FormControl](#) formContrl,

[FS\\_FloatRect](#)\* outRect

```
);
```

### Description

Gets the rectangle of the widget.

### Parameter

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

outRect	[Out] The rectangle of the widget.
---------	------------------------------------

---

### Return

void

### Head file reference

fpd\_docTempl.h: 5550

## FPDFFormControlGetRolloverCaption

### Syntax

```
void FPDFFormControlGetRolloverCaption (
```

[FPD\\_FormControl](#) formContrl,

[FS\\_WideString](#)\* outCaption

```
);
```

### Description

Gets the widget annotation's rollover caption.

### Parameter

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

outCaption	[Out] The widget annotation's rollover caption.
------------	---

---

### Return

void

### Head file reference

fpd\_docTempl.h: 5795



## FPDFFormControlGetRolloverIcon

### Syntax

```
FPD_Object FPDFFormControlGetRolloverIcon (
    FPD FormControl formContrl
);
```

### Description

Gets the widget annotation's rollover icon.

### Parameter

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

### Return

The widget annotation's rollover icon.

### Head file reference

fpd\_docTempl.h: 5874

## FPDFFormControlGetRotation

### Syntax

```
FS_INT32 FPDFFormControlGetRotation (
    FPD FormControl formContrl
);
```

### Description

Gets the number of degrees by which the widget annotation is rotated counterclockwise relative to the page. The value must be a multiple of 90.

### Parameter

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

### Return

The number of degrees.

### Head file reference

fpd\_docTempl.h: 5661

## FPDFFormControlGetPosition

### Syntax

```
FS_INT32 FPDFFormControlGetPosition (
    FPD FormControl formContrl
);
```



**Description**

Gets where to position the text of the widget annotation's caption relative to its icon.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

**Return**

The text position.

**Head file reference**

fpd\_docTempl.h: 5934

**FPDFormControlGetType****Syntax**

```
FPD_FormFieldType FPDFormControlGetType (
    FPD FormControl formContrl
);
```

**Description**

Gets the field type.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

**Return**

A PDF interactive form control.

**Head file reference**

fpd\_docTempl.h: 5514

**FPDFormControlGetWidget****Syntax**

```
FPD_Object FPDFormControlGetWidget (
    FPD FormControl formContrl
);
```

**Description**

Gets the widget annotation dictionary.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

**Return**

The widget annotation dictionary.

**Head file reference**

fpd\_docTempl.h: 5541

**FPDFormControlHasMKEntry****Syntax**

```
FS_BOOL FPDFormControlHasMKEntry (
    FPD FormControl formContrl,
    FS LPSTR szEntry
);
```

**Description**

Checks whether the specified entry exist in the appearance characteristics dictionary.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

---

szEntry	[In] The input entry name.
---------	----------------------------

---

**Return**

Non-zero means exist, otherwise not exist.

**Head file reference**

fpd\_docTempl.h: 5641

**FPDFormControlIsChecked****Syntax**

```
FS_BOOL FPDFormControlIsChecked (
    FPD FormControl formContrl
);
```

**Description**

Checks whether the control is checked.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

**Return**

[TRUE](#) for the control being checked.

**Head file reference**

fpd\_docTempl.h: 5604

**FPDFFormControlIsDefaultChecked****Syntax**

```
FS_BOOL FPDFFormControlIsDefaultChecked (
    FPDFormControl formContl
);
```

**Description**

Checks whether the control is default checked.

**Parameter**

---

formContl	[In] The input PDF interactive form control.
-----------	--

---

**Return**

[TRUE](#) for the contro being default checked.

**Head file reference**

fpd\_docTempl.h: 5613

**FPDFFormControlRemoveMKEntry****Syntax**

```
void FPDFFormControlRemoveMKEntry (
    FPDFormControl formContl,
    FS\_LPSTR szEntry
);
```

**Description**

Removes a entry in the appearance characteristics dictionary.

**Parameter**

---

formContl	[In] The input PDF interactive form control.
-----------	--

---

---

szEntry	[In] The input entry name to be removed.
---------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5651

## FPDFFormControlSetAction

### Syntax

```
void FPDFFormControlSetAction (
    FPDFormControl formContrl,
    const FPD Action action
);
```

### Description

Sets the action to be performed when the annotation is activated.

### Parameter

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

action	[In] The input action.
--------	------------------------

---

### Return

void

### Head file reference

fpd\_docTempl.h: 5963

## FPDFFormControlSetAdditionalAction

### Syntax

```
void FPDFFormControlSetAdditionalAction (
    FPD FormControl formContrl,
    const FPD AAction aaction
);
```

### Description

Sets the additional-actions.

### Parameter

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

aaction	[In] The input additional-actions.
---------	------------------------------------

---

### Return

void



**Head file reference**

fpd\_docTempl.h: 5983

**FPDFFormControlSetBackgroundColor****Syntax**

```
void FPDFFormControlSetBackgroundColor (
    FPD\_FormControl formContrl,
    FS\_INT32 iColorType,
    FS\_ARGB color
);
```

**Description**

Sets the background color.

**Parameter**

formContrl	[In] The input PDF interactive form control.
iColorType	[In] The input color type.
color	[In] The FS_ARGB value of the input color.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5754

**FPDFFormControlSetBorderColor****Syntax**

```
void FPDFFormControlSetBorderColor (
    FPD\_FormControl formContrl,
    FS\_INT32 iColorType,
    FS\_ARGB color
);
```

**Description**

Sets the border color.

**Parameter**

formContrl	[In] The input PDF interactive form control.
------------	--



iColorType	[In] The input color type.
------------	----------------------------

color	[In] The FS_ARGB value of the input color.
-------	--

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5712

**FPDFormControlSetControlAlignment****Syntax**

```
void FPDFormControlSetControlAlignment (
    FPDFormControl formContrl,
    FS\_INT32 iAlignment
);
```

**Description**

Sets the alignment of the control.

**Parameter**

formContrl	[In] The input PDF interactive form control.
------------	--

iAlignment	[In] The input alignment. 0 - left alignment, the default setting. 1 - centered, 2 - right alignment.
------------	--

**Return**

void.

**Head file reference**

fpd\_docTempl.h: 6045

**FPDFormControlSetDefaultAppearance****Syntax**

```
void FPDFormControlSetDefaultAppearance (
    FPDFormControl formContrl,
    const FPD\_DefaultAppearance cDA
);
```

**Description**

Sets the default appearance.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

cDA	[In] The input default appearance.
-----	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 6004

**FPDFFormControlSetDefaultControlFont****Syntax**

```
void FPDFFormControlSetDefaultControlFont (
    FPDFormControl formContrl,
    const FPD Font font
);
```

**Description**

Sets the default font of the control.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

font	[In] The input PDF font.
------	--------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 6023

**FPDFFormControlSetDownCaption****Syntax**

```
void FPDFFormControlSetDownCaption (
    FPD FormControl formContrl,
    FS LPCSTR szCaption
);
```

**Description**

Sets the widget annotation's down caption. ANSI version.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

szCaption	[In] The input down caption.
-----------	------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5835

**FPDFFormControlSetDownCaptionW****Syntax**

```
void FPDFFormControlSetDownCaptionW (
    FPD\_FormControl formContrl,
    FS\_LPCWSTR wszCaption
);
```

**Description**

Sets the widget annotation's down caption. Unicode version.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

wszCaption	[In] The input down caption.
------------	------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5845

**FPDFFormControlSetDownIcon****Syntax**

```
void FPDFFormControlSetDownIcon (
    FPD\_FormControl formContrl,
    const FPD\_Object iconStream
);
```

**Description**

Sets the widget annotation's down icon.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
iconStream	[In] The input PDF stream of down icon.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5902

**FPDFFormControlSetExportValue****Syntax**

```
void FPDFFormControlSetExportValue (
    FPDFormControl formContrl,
    FS\_LPCWSTR wszValue,
    FS\_BOOL bNotify
);
```

**Description**

Sets the export mapping name.

**Parameter**


---

formContrl	[In] The input PDF interactive form control.
wszValue	[In] The input export name.
bNotify	[In] Whether do notifying.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5593

**FPDFFormControlSetHighlightingMode****Syntax**

```
void FPDFFormControlSetHighlightingMode (
    FPDFormControl formContrl,
    FPDFormCtrlHighlightingMode eHLMMode
);
```

**Description**

Sets the highlighting mode.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

eHLMMode	[In] The input highlighting mode.
----------	-----------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5631

**FPDFormControlSetIconFit****Syntax**

```
void FPDFormControlSetIconFit (
    FPD FormControl formContrl,
    const FPD IconFit cIF,
    void* pObjs
);
```

**Description**

Sets the icon fit of the widget.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

cIF	[In] The input icon fit.
-----	--------------------------

---

pObjs	[In] The indirect object collection.
-------	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5922

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDFormControlSetNormalCaption****Syntax**

```
void FPDFFormControlSetNormalCaption (
```

FPD\_FormControl formContrl,

FS\_LPCSTR szCaption

```
);
```

**Description**

Sets the widget annotation's normal caption. ANSI version.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

szCaption	[In] The input normal caption.
-----------	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5775

**FPDFFormControlSetNormalCaptionW****Syntax**

```
void FPDFFormControlSetNormalCaptionW (
```

FPD\_FormControl formContrl,

FS\_LPCWSTR wszCaption

```
);
```

**Description**

Sets the widget annotation's normal caption. Unicode version.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

wszCaption	[In] The input normal caption.
------------	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5785

**FPDFFormControlSetNormalIcon**

**Syntax**

```
void FPDFormControlSetNormalIcon (
    FPD FormControl formContrl,
    const FPD Object iconStm
);
```

**Description**

Sets the widget annotation's normal icon.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

iconStm	[In] The input PDF stream of normal icon.
---------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5864

**FPDFormControlSetRolloverCaption****Syntax**

```
void FPDFormControlSetRolloverCaption (
    FPD FormControl formContrl,
    FS LPCSTR szCaption
);
```

**Description**

Sets the widget annotation's rollover caption. ANSI version.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

szCaption	[In] The input rollover caption.
-----------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5805

**FPDFormControlSetRolloverCaptionW**

**Syntax**

```
void FPDFormControlSetRolloverCaptionW (
    FPD FormControl formContrl,
    FS LPCWSTR wszCaption
);
```

**Description**

Sets the widget annotation's rollover caption. Unicode version.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

wszCaption	[In] The input rollover caption.
------------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5815

**FPDFormControlSetRolloverIcon****Syntax**

```
void FPDFormControlSetRolloverIcon (
    FPD FormControl formContrl,
    const FPD Object iconStream
);
```

**Description**

Sets the widget annotation's rollover icon.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

iconStream	[In] The input PDF stream of rollover icon.
------------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5883

**FPDFormControlSetRotation**

**Syntax**

```
void FPDFormControlSetRotation (
    FPD FormControl formContrl,
    FS\_INT32 iDegree
);
```

**Description**

Sets the rotation.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

iDegree	[In] The input number of degrees to rotated counterclockwise.
---------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5671

**FPDFormControlSetTextPosition****Syntax**

```
void FPDFormControlSetTextPosition (
    FPD FormControl formContrl,
    FS\_INT32 iPos
);
```

**Description**

Sets the text position.

**Parameter**

---

formContrl	[In] The input PDF interactive form control.
------------	--

---

iPos	[In] The input text position.
------	-------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5943

**FPD\_FormField**

## [Return from Used by](#)

### Description

A field in a interactive form. See [FPDInterFormNewField](#) , [FPDInterFormGetField](#) , [FPDInterFormGetFieldByDict](#) , [FPDInterFormDeleteField](#) , [FPDInterFormDeleteField2](#) .

### Returned from

[FPDFFormControlGetField](#)  
[FPDFFormControlGetType](#)  
[FPDInterFormCheckRequiredFields](#)  
[FPDInterFormGetField](#)  
[FPDInterFormGetFieldByDict](#)  
[FPDInterFormGetFieldInCalculationOrder](#)  
[FPDInterFormNewField](#)  
[FPDFFormFieldGetType](#)

### Used by

[FPDInterFormDeleteField2](#)  
[FPDInterFormRenameField2](#)  
[FPDFFormFieldCheckControl](#)  
[FPDFFormFieldClearOptions](#)  
[FPDFFormFieldClearSelectedOptions](#)  
[FPDFFormFieldClearSelection](#)  
[FPDFFormFieldCountControls](#)  
[FPDFFormFieldCountOptions](#)  
[FPDFFormFieldCountSelectedItems](#)  
[FPDFFormFieldCountSelectedOptions](#)  
[FPDFFormFieldDefaultCheckControl](#)  
[FPDFFormFieldDeleteOption](#)  
[FPDFFormFieldFindOption](#)  
[FPDFFormFieldFindOptionValue](#)  
[FPDFFormFieldGetAdditionalAction](#)  
[FPDFFormFieldGetAlternateName](#)  
[FPDFFormFieldGetControl](#)  
[FPDFFormFieldGetControlItemIndex](#)  
[FPDFFormFieldGetDefaultSelectedItem](#)  
[FPDFFormFieldGetDefaultCellStyle](#)  
[FPDFFormFieldGetDefaultValue](#)  
[FPDFFormFieldGetFieldDict](#)  
[FPDFFormFieldGetFieldFlags](#)  
[FPDFFormFieldGetFieldType](#)  
[FPDFFormFieldGetFlags](#)  
[FPDFFormFieldGetFont](#)  
[FPDFFormFieldGetFontSize](#)  
[FPDFFormFieldGetFullName](#)  
[FPDFFormFieldGetInterForm](#)  
[FPDFFormFieldGetMappingName](#)  
[FPDFFormFieldGetMaxLen](#)  
[FPDFFormFieldGetOptionLabel](#)  
[FPDFFormFieldGetOptionValue](#)  
[FPDFFormFieldGetRichTextString](#)  
[FPDFFormFieldGetSelectedIndex](#)

[FPDFFormFieldGetSelectedOptionIndex](#)  
[FPDFFormFieldGetTopVisibleIndex](#)  
[FPDFFormFieldGetType](#)  
[FPDFFormFieldGetValue](#)  
[FPDFFormFieldInsertOption](#)  
[FPDFFormFieldIsItemDefaultSelected](#)  
[FPDFFormFieldIsItemSelected](#)  
[FPDFFormFieldIsOptionSelected](#)  
[FPDFFormFieldResetField](#)  
[FPDFFormFieldSelectOption](#)  
[FPDFFormFieldSetAdditionalAction](#)  
[FPDFFormFieldSetAlternateName](#)  
[FPDFFormFieldSetAlternateNameW](#)  
[FPDFFormFieldSetDefaultStyle](#)  
[FPDFFormFieldSetDefaultValue](#)  
[FPDFFormFieldSetFieldFlags](#)  
[FPDFFormFieldSetItemDefaultSelection](#)  
[FPDFFormFieldSetItemSelection](#)  
[FPDFFormFieldSetMappingName](#)  
[FPDFFormFieldSetMappingNameW](#)  
[FPDFFormFieldSetMaxLen](#)  
[FPDFFormFieldSetOptionLabel](#)  
[FPDFFormFieldSetOptionValue](#)  
[FPDFFormFieldSetRichTextString](#)  
[FPDFFormFieldSetRichTextStringW](#)  
[FPDFFormFieldSetTopVisibleIndex](#)  
[FPDFFormFieldSetValue](#)  
[FPDFFormFieldUpdateUnisonStatus](#)

## Definitions

### Definitions summary

#### [FPD FORM COMBO EDIT](#)

(Meaningful only when FORMCHOICE\_COMBO flag is set): if set, the combo box includes an editable text control, otherwise, it's only a drop list.

#### [FPD FORM FIELD NOEXPORT](#)

Don't export the field when submitting. Not supported for now.

#### [FPD FORM FIELD READONLY](#)

This field is read only, no editing allowed.

#### [FPD FORM FIELD REQUIRED](#)

This field is required when submit.

#### [FPD TEXT POS ABOVE](#)

caption above the icon.

#### [FPD TEXT POS BELOW](#)

caption below the icon.

#### [FPD TEXT POS CAPTION](#)

no icon, caption only.

#### [FPD TEXT POS ICON](#)

no caption, icon only.

#### [FPD TEXT POS LEFT](#)

caption to the left of the icon.

#### [FPD TEXT POS OVERLAI](#)



**FPD\_TEXT\_POS\_RIGHT**

caption to the right of the icon.

**FPD\_FORM\_LIST\_MULTISELECT**

If set, more than one items can be selected.

**FPD\_FORM\_RADIO\_NOTOGGLEOFF**

For radio button only: if set, one radio button has to be selected at any time; Otherwise, when the selected radio button is clicked, it will be turned off, (leaving no radio button selected at this time).

**FPD\_FORM\_RADIO\_UNISON**

If set, radio buttons with same value in a field will be turned on or off unison (either all one, or all off). If cleared, all buttons are usually exclusive.

**FPD\_FORM\_TEXT\_COMB**

If set, this field is arranged in a number of equally spaced positions ("combs"), the number of positions is determined by MaxLen parameter.

**FPD\_FORM\_TEXT\_MULTILINE**

Multiple lines.

**FPD\_FORM\_TEXT\_NOSCROLL**

Do not scroll (vertically for multi-line field, or horizontally for single-line field). If the field is full, no more text is accepted.

**FPD\_FORM\_TEXT\_PASSWORD**

This is a password field. Password shouldn't be displayed or exported.

## Definitions detail

### FPD\_FORM\_COMBO\_EDIT

**Syntax**

```
#define FPD_FORM_COMBO_EDIT 0x100
```

**Description**

(Meaningful only when FORMCHOICE\_COMBO flag is set): if set, the combo box includes an editable text control, otherwise, it's only a drop list.

**Group**

[FPDComboBoxFieldsAdditionalFlags](#)

**Head file reference**

fpd\_docExpT.h: 907

### FPD\_FORM\_FIELD\_NOEXPORT

**Syntax**

```
#define FPD_FORM_FIELD_NOEXPORT 0x04
```

**Description**

Don't export the field when submitting. Not supported for now.

**Group**

[FPDFormFieldFlags](#)

**Head file reference**

fpd\_docExpT.h: 840

**FPD\_FORM\_FIELD\_READONLY****Syntax**

#define FPD\_FORM\_FIELD\_READONLY 0x01

**Description**

This field is read only, no editing allowed.

**Group**[FPDFFormFieldFlags](#)**Head file reference**

fpd\_docExpT.h: 836

**FPD\_FORM\_FIELD\_REQUIRED****Syntax**

#define FPD\_FORM\_FIELD\_REQUIRED 0x02

**Description**

This field is required when submit.

**Group**[FPDFFormFieldFlags](#)**Head file reference**

fpd\_docExpT.h: 838

**FPD\_TEXT\_POS\_ABOVE****Syntax**

#define FPD\_TEXT\_POS\_ABOVE 3

**Description**

caption above the icon.

**Group**[FPDFFormFieldTextPosition](#)**Head file reference**

fpd\_docExpT.h: 982



## FPD\_TEXT\_POS\_BELOW

### Syntax

```
#define FPD_TEXT_POS_BELOW 2
```

### Description

caption below the icon.

### Group

[FPDFormFieldTextPosition](#)

### Head file reference

fpd\_docExpT.h: 980

## FPD\_TEXT\_POS\_CAPTION

### Syntax

```
#define FPD_TEXT_POS_CAPTION 0
```

### Description

no icon, caption only.

### Group

[FPDFormFieldTextPosition](#)

### Head file reference

fpd\_docExpT.h: 976

## FPD\_TEXT\_POS\_ICON

### Syntax

```
#define FPD_TEXT_POS_ICON 1
```

### Description

no caption, icon only.

### Group

[FPDFormFieldTextPosition](#)

### Head file reference

fpd\_docExpT.h: 978

## FPD\_TEXT\_POS\_LEFT

### Syntax

```
#define FPD_TEXT_POS_LEFT 5
```



**Description**

caption to the left of the icon.

**Group**

[FPDFormFieldTextPosition](#)

**Head file reference**

fpd\_docExpT.h: 986

## FPD\_TEXT\_POS\_OVERLAID

**Syntax**

#define FPD\_TEXT\_POS\_OVERLAID 6

**Description****Group**

[FPDFormFieldTextPosition](#)

**Head file reference**

fpd\_docExpT.h: 988

## FPD\_TEXT\_POS\_RIGHT

**Syntax**

#define FPD\_TEXT\_POS\_RIGHT 4

**Description**

caption to the right of the icon.

**Group**

[FPDFormFieldTextPosition](#)

**Head file reference**

fpd\_docExpT.h: 984

## FPD\_FORM\_LIST\_MULTISELECT

**Syntax**

#define FPD\_FORM\_LIST\_MULTISELECT 0x100

**Description**

If set, more than one items can be selected.

**Group**

[FPDListBoxFieldsAdditionalFlags](#)

**Head file reference**

fpd\_docExpT.h: 921

**FPD\_FORM\_RADIO\_NOTOGGLEOFF****Syntax**

#define FPD\_FORM\_RADIO\_NOTOGGLEOFF 0x100

**Description**

For radio button only: if set, one radio button has to be selected at any time; Otherwise, when the selected radio button is clicked, it will be turned off, (leaving no radio button selected at this time).

**Group**[FPDRadioButtonAdditionalFlags](#)**Head file reference**

fpd\_docExpT.h: 858

**FPD\_FORM\_RADIO\_UNISON****Syntax**

#define FPD\_FORM\_RADIO\_UNISON 0x200

**Description**

If set, radio buttons with same value in a field will be turned on or off unison (either all one, or all off). If cleared, all buttons are usually exclusive.

**Group**[FPDRadioButtonAdditionalFlags](#)**Head file reference**

fpd\_docExpT.h: 864

**FPD\_FORM\_TEXT\_COMB****Syntax**

#define FPD\_FORM\_TEXT\_COMB 0x800

**Description**

If set, this field is arranged in a number of equally spaced positions ("combs"), the number of positions is determined by MaxLen parameter.

**Group**[FPDTTextFieldsAdditionalFlags](#)**Head file reference**

fpd\_docExpT.h: 890

## FPD\_FORM\_TEXT\_MULTILINE

### Syntax

#define FPD\_FORM\_TEXT\_MULTILINE 0x100

### Description

Multiple lines.

### Group

[FPDTextFieldsAdditionalFlags](#)

### Head file reference

fpd\_docExpT.h: 878

## FPD\_FORM\_TEXT\_NOSCROLL

### Syntax

#define FPD\_FORM\_TEXT\_NOSCROLL 0x400

### Description

Do not scroll (vertically for multi-line field, or horizontally for single-line field). If the field is full, no more text is accepted.

### Group

[FPDTextFieldsAdditionalFlags](#)

### Head file reference

fpd\_docExpT.h: 885

## FPD\_FORM\_TEXT\_PASSWORD

### Syntax

#define FPD\_FORM\_TEXT\_PASSWORD 0x200

### Description

This is a password field. Password shouldn't be displayed or exported.

### Group

[FPDTextFieldsAdditionalFlags](#)

### Head file reference

fpd\_docExpT.h: 880

## Enumerations



## Enumerations summary

### [\*\*FPD\\_FormCtrlHighlightingMode\*\*](#)

HighLighting mode definition, for annotation or control. See [FPDFControlGetHighlightingMode](#) .

### [\*\*FPD\\_FormFieldType\*\*](#)

PDF interactive form field type enumeration. See [FPDFFormFieldGetType](#) .

### [\*\*FPD\\_IconScaleMethod\*\*](#)

Scale method enumeration. The circumstances under which the icon should be scaled inside the annotation rectangle. See [FPDIIconFitSetScaleMethod](#) .

## Enumerations detail

### FPD\_FormCtrlHighlightingMode

#### **Syntax**

```
enum FPD_FormCtrlHighlightingMode{  
    FormCtrlHL_None,  
    FormCtrlHL_Invert,  
    FormCtrlHL_Outline,  
    FormCtrlHL_Push,  
    FormCtrlHL_ToggleMode  
};
```

#### **Description**

HighLighting mode definition, for annotation or control. See [FPDFControlGetHighlightingMode](#) .

#### **Head file reference**

fpd\_docExpT.h: 1001

##### **FormCtrlHL\_None**

(None) No highlighting.

##### **FormCtrlHL\_Invert**

(Invert) Invert the contents of the annotation rectangle.

##### **FormCtrlHL\_Outline**

(Outline) Invert the annotation's border.

##### **FormCtrlHL\_Push**

(Push) Display the annotation's down appearance, if any;If no down appearance is defined, offset the contents of the annotation rectangle to appear as if it were being pushed below the surface of the page.

##### **FormCtrlHL\_ToggleMode**

(Toggle) Same as P (which is preferred).

### FPD\_FormFieldType

#### **Syntax**

```
enum FPD_FormFieldType{
```



```
FormField_Uncertain,  
FormField_PushButton,  
FormField_RadioButton,  
FormField_CheckBox,  
FormField_TextField,  
FormField_RichText,  
FormField_File,  
FormField_ListBox,  
FormField_ComboBox,  
FormField_Sign  
};
```

**Description**

PDF interactive form field type enumeration. See [FPDFormFieldGetType](#) .

**Head file reference**

fpd\_docExpT.h: 934

**FormField\_Uncertain**

unsupported or invalid field.

**FormField\_PushButton**

push button without any value.

**FormField\_RadioButton**

a group of radio buttons, at most one item can be selected.

**FormField\_CheckBox**

check box with on/off states.

**FormField\_TextField**

single line or multi-line texts.

**FormField\_RichText**

rich text (using XML to store rich text).

**FormField\_File**

file content.

**FormField\_ListBox**

list box, single or multiple selections.

**FormField\_ComboBox**

combo box with or without the edit box.

**FormField\_Sign**

digital signature.

**FPD\_IconScaleMethod****Syntax**

```
enum FPD_IconScaleMethod{
    IconScaleMethod_Always,
    IconScaleMethod_Bigger,
    IconScaleMethod_Smaller,
    IconScaleMethod_Never
};
```

### Description

Scale method enumeration. The circumstances under which the icon should be scaled inside the annotation rectangle. See [FPDIconFitSetScaleMethod](#) .

### Head file reference

fpd\_docExpT.h: 958

#### **IconScaleMethod\_Always**

A, Always scale.

#### **IconScaleMethod\_Bigger**

B, Scale only when the icon is bigger than the annotation rectangle.

#### **IconScaleMethod\_Smaller**

S, Scale only when the icon is smaller then the annotation rectangle.

#### **IconScaleMethod\_Never**

N, Never scale.

## Functions

### Functions summary

#### [\*\*FPDFFormFieldCheckControl\*\*](#)

Checks a control.

#### [\*\*FPDFFormFieldClearOptions\*\*](#)

Clears all options.

#### [\*\*FPDFFormFieldClearSelectedOptions\*\*](#)

Clears selected options to be unselected.

#### [\*\*FPDFFormFieldClearSelection\*\*](#)

Clears the selection.

#### [\*\*FPDFFormFieldCountControls\*\*](#)

Gets the count of controls belongs to this field.

#### [\*\*FPDFFormFieldCountOptions\*\*](#)

Gets the count of options.

#### [\*\*FPDFFormFieldCountSelectedItems\*\*](#)

Gets the count of selected items.

#### [\*\*FPDFFormFieldCountSelectedOptions\*\*](#)

Gets the count of option indices in the "I" entry list.

#### [\*\*FPDFFormFieldDefaultCheckControl\*\*](#)

Checks a control's default check state.

#### [\*\*FPDFFormFieldDeleteOption\*\*](#)

Deletes an option.

**FPDFFormFieldFindOption**

Finds an option by the label.

**FPDFFormFieldFindOptionValue**

Finds an option by it's value.

**FPDFFormFieldGetAdditionalAction**

Gets the additional action of the field.

**FPDFFormFieldGetAlternateName**

Gets the alternate field name to be used in place of the actual field name wherever the field must be identified in the user interface.

**FPDFFormFieldGetControl**

Gets a control.

**FPDFFormFieldGetControlIndex**

Gets the index of a control.

**FPDFFormFieldGetDefaultSelectedItem**

Gets the default selected item index.

**FPDFFormFieldGetDefaultCellStyle**

Gets the default style string.

**FPDFFormFieldGetDefaultValue**

Gets the default value of the field.

**FPDFFormFieldGetFieldDict**

Gets the field dictionary.

**FPDFFormFieldGetFieldFlags**

Gets the field flags.

**FPDFFormFieldGetFieldType**

Gets the field type.

**FPDFFormFieldGetFlags**

Gets the field flags.

**FPDFFormFieldGetFont**

Gets the font.

**FPDFFormFieldGetFontSize**

Gets the font size.

**FPDFFormFieldGetFullName**

Gets the full name of the field.

**FPDFFormFieldGetInterForm**

Gets the interactive form which it belongs to.

**FPDFFormFieldGetMappingName**

Gets the mapping name to be used when exporting interactive form field data from the document.

**FPDFFormFieldGetMaxLen**

Gets the maximum length of the field's text, in characters.

**FPDFFormFieldGetOptionLabel**

Gets the label of specified option.

**FPDFFormFieldGetOptionValue**

Gets the value of specified option.

**FPDFFormFieldGetRichTextString**

Gets the rich text string.

**FPDFFormFieldGetSelectedIndex**

Gets the selected item index in the item array(include selected and unselected).

**FPDFFormFieldGetSelectedOptionIndex**

Gets the index of a selection option in all option array.

**FPDFFormFieldGetTopVisibleIndex**

Gets the top visible option index.

**FPDFFormFieldGetType**

Gets the type of the field.

**FPDFFormFieldGetValue**

Gets the value of the field.

**FPDFFormFieldInsertOption**

Inserts an option at specified position.

**FPDFFormFieldIsItemDefaultSelected**

Checks whether the specified item's default selection flag is set.

**FPDFFormFieldIsItemSelected**

Checks whether the specified item has been selected.

**FPDFFormFieldIsOptionSelected**

Checks whether specified option has been selected.

**FPDFFormFieldResetField**

Resets the form field.

**FPDFFormFieldSelectOption**

Sets specified option to be selected.

**FPDFFormFieldSetAdditionalAction**

Sets the additional action of the field.

**FPDFFormFieldSetAlternateName**

Sets the alternate field name. ANSI version.

**FPDFFormFieldSetAlternateNameW**

Sets the alternate field name. Unicode version.

**FPDFFormFieldSetDefaultStyle**

Sets the default style string.

**FPDFFormFieldSetDefaultValue**

Sets the default value of the field.

**FPDFFormFieldSetFieldFlags**

Sets the field flags.

**FPDFFormFieldSetItemDefaultSelection**

Sets the default selection flag of specified item.

**FPDFFormFieldSetItemSelection**

Sets the selection flag of specified item.

**FPDFFormFieldSetMappingName**

Sets the mapping name. ANSI version.

**FPDFFormFieldSetMappingNameW**

Sets the mapping name. Unicode version.

**FPDFFormFieldSetMaxLen**

Sets the maximum length of the field's text, in characters.

**FPDFFormFieldSetOptionLabel**

Sets the option label.

**FPDFFormFieldSetOptionValue**

Sets the option value.

**FPDFFormFieldSetRichTextString**

Sets the rich text string. ANSI version.

**FPDFFormFieldSetRichTextStringW**

Sets the rich text string. Unicode version.

**FPDFFormFieldSetTopVisibleIndex**

Sets the top visible option index.

**FPDFFormFieldSetValue**



Sets the value of the field. not applicable to non-unison radio box.

#### **FPDFormFieldUpdateUnisonStatus**

Resets or recreate Opt array list and AP stream, for RadioButton only. This is used whenever unison status is changed.

### Functions detail

#### FPDFormFieldCheckControl

##### **Syntax**

```
FS_BOOL FPDFormFieldCheckControl (
    FPDFormField formField,
    FS_INT32 iControlIndex,
    FS_BOOL bChecked,
    FS_BOOL bNotify
);
```

##### **Description**

Checks a control.

##### **Parameter**

formField	[In] The input PDF interactive form field.
iControlIndex	[In] The index of the control.
bChecked	[In] The input check state.
bNotify	[In] Whether do notifying.

##### **Return**

Non-zero means success, otherwise failure.

##### **Head file reference**

fpd\_docTempl.h: 5267

#### FPDFormFieldClearOptions

##### **Syntax**

```
FS_BOOL FPDFormFieldClearOptions (
    FPDFormField formField,
    FS_BOOL bNotify
);
```

##### **Description**

Clears all options.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

bNotify	[In] Whether do notifying.
---------	----------------------------

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 5257

**FPDFormFieldClearSelectedOptions****Syntax**

```
FS_BOOL FPDFormFieldClearSelectedOptions (
    FPD\_FormField formField,
    FS\_BOOL bNotify
);
```

**Description**

Clears selected options to be unselected.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

bNotify	[In] Whether do notifying.
---------	----------------------------

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 5361

**FPDFormFieldClearSelection****Syntax**

```
FS_BOOL FPDFormFieldClearSelection (
    FPD\_FormField formField,
    FS\_BOOL bNotify
);
```

**Description**

Clears the selection.



**Parameter**

---

formField	[In] The input PDF interactive form field.
bNotify	[In] Whether do notifying.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 5096

**FPDFormFieldCountControls****Syntax**

```
FS_INT32 FPDFormFieldCountControls (
    FPDFormField formField
);
```

**Description**

Gets the count of controls belongs to this field.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

**Return**

The count of controls belongs to this field.

**Head file reference**

fpd\_docTempl.h: 4801

**FPDFormFieldCountOptions****Syntax**

```
FS_INT32 FPDFormFieldCountOptions (
    FPDFormField formField
);
```

**Description**

Gets the count of options.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

**Return**

The count of options.

**Head file reference**

fpd\_docTempl.h: 5158

**FPDFFormFieldCountSelectedItems****Syntax**

```
FS_INT32 FPDFFormFieldCountSelectedItems (
    FPDFormField formField
);
```

**Description**

Gets the count of selected items.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

**Return**

The count of selected items.

**Head file reference**

fpd\_docTempl.h: 5077

**FPDFFormFieldCountSelectedOptions****Syntax**

```
FS_INT32 FPDFFormFieldCountSelectedOptions (
    FPDFormField formField
);
```

**Description**

Gets the count of option indices in the "I" entry list.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

**Return**

The count of option indices in the "I" entry list.

**Head file reference**

fpd\_docTempl.h: 5320

## FPDFormFieldDefaultCheckControl

### Syntax

```
FS_BOOL FPDFormFieldDefaultCheckControl (
    FPD_FormField formField,
    FS_INT32 iControlIndex,
    FS_BOOL bChecked
);
```

### Description

Checks a control's default check state.

### Parameter

---

formField	[In] The input PDF interactive form field.
-----------	--

---

iControlIndex	[In] The index of the control.
---------------	--------------------------------

---

bChecked	[In] the input check state.
----------	-----------------------------

---

### Return

Non-zero means success, otherwise failure.

### Head file reference

fpd\_docTempl.h: 5279

## FPDFormFieldDeleteOption

### Syntax

```
FS_BOOL FPDFormFieldDeleteOption (
    FPD_FormField formField,
    FS_INT32 index,
    FS_BOOL bNotify
);
```

### Description

Deletes an option.

### Parameter

---

formField	[In] The input PDF interactive form field.
-----------	--

---

index	[In] The zero-based index of the option to be deleted.
-------	--

---



---

bNotify	[In] Whether do notifying.
---------	----------------------------

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 5246

**FPDFormFieldFindOption****Syntax**

```
FS_INT32 FPDFormFieldFindOption (
    FPDFormField formField,
    FS\_LPWSTR wszOptLabel
);
```

**Description**

Finds an option by the label.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

---

wszOptLabel	[In] The label of the option to be found.
-------------	---

---

**Return**

The index of the found option.

**Head file reference**

fpd\_docTempl.h: 5201

**FPDFormFieldFindOptionValue****Syntax**

```
FS_INT32 FPDFormFieldFindOptionValue (
    FPDFormField formField,
    FS\_LPCWSTR csOptValue,
    FS\_INT32 istartIndex
);
```

**Description**

Finds an option by it's value.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---



---

csOptValue	[In] The value of the option to be found.
------------	---

---

istartIndex	[In] The start option index to find.
-------------	--------------------------------------

---

**Return**

The index of the found option.

**Head file reference**

fpd\_docTempl.h: 5211

**FPDFormFieldGetAdditionalAction****Syntax**

```
void FPDFormFieldGetAdditionalAction (
    FPD\_FormField formField,
    FPD\_AAction* outAAction
);
```

**Description**

Gets the additional action of the field.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

outAAction	[Out] The additional action of the field.
------------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4839

**FPDFormFieldGetAlternateName****Syntax**

```
void FPDFormFieldGetAlternateName (
    FPD\_FormField formField,
    FS\_WideString* outName
);
```

**Description**

Gets the alternate field name to be used in place of the actual field name wherever the field must be identified in the user interface.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

outName	[Out] The alternate field name
---------	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4859

**FPDFormFieldGetControl****Syntax**

```
FPD_FormControl FPDFormFieldGetControl (
    FPD\_FormField formField,
    FS\_INT32 index
);
```

**Description**

Gets a control.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

index	[In] The input zero-based control index in the field.
-------	---

---

**Return**

A form control.

**Head file reference**

fpd\_docTempl.h: 4810

**FPDFormFieldGetControlIndex****Syntax**

```
FS_INT32 FPDFormFieldGetControlIndex (
    FPD\_FormField formField,
    FPD\_FormControl control
);
```

**Description**

Gets the index of a control.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

control	[In] The input form control.
---------	------------------------------

---

**Return**

The index of the control in the field.

**Head file reference**

fpd\_docTempl.h: 4820

**FPDFormFieldGetDefaultSelectedItem****Syntax**

```
FS_INT32 FPDFormFieldGetDefaultSelectedItem (
    FPD\_FormField formField
);
```

**Description**

Gets the default selected item index.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

**Return**

The default selected item index.

**Head file reference**

fpd\_docTempl.h: 5149

**FPDFormFieldGetDefaultStyle****Syntax**

```
void FPDFormFieldGetDefaultStyle (
    FPD\_FormField formField,
    FS\_ByteString* outStyle
);
```

**Description**

Gets the default style string.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---



---

outStyle	[Out] The default style string.
----------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4939

**FPDFormFieldGetDefaultValue****Syntax**

```
void FPDFormFieldGetDefaultValue (
    FPD_FormField formField,
    FS_WideString* outValue
);
```

**Description**

Gets the default value of the field.

**Parameter**


---

formField	[In] The input PDF interactive form field.
-----------	--

---



---

outValue	[Out] The default value of the field.
----------	---------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5006

**Note:**

- For a text field, the value is the text string;
- For a rich text field, the value is the rich text XML element;
- For a file field, the value is the path name of the file;
- For a radio button or check box field, the value is value string of the selected button;
- For a list box field, the value is the value of first selected item, if any;
- For a comb box field, the value is the text string.

**FPDFormFieldGetFieldDict****Syntax**

```
FPD_Object FPDFormFieldGetFieldDict (
    FPD_FormField formField
);
```



**Description**

Gets the field dictionary.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

**Return**

The field dictionary.

**Head file reference**

fpd\_docTempl.h: 4782

**FPDFormFieldGetFieldFlags****Syntax**

```
FS_DWORD FPDFormFieldGetFieldFlags (
    FPDFormField formField
);
```

**Description**

Gets the field flags.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

**Return**

The field flags.

**Head file reference**

fpd\_docTempl.h: 4920

**FPDFormFieldGetFieldType****Syntax**

```
FS_INT32 FPDFormFieldGetFieldType (
    FPDFormField formField
);
```

**Description**

Gets the field type.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

**Return**

Types like FPD\_FORM\_FIELDTYPE\_xxxx

**Head file reference**

fpd\_docTempl.h: 4830

**FPDFFormFieldGetFlags****Syntax**

```
FS_DWORD FPDFFormFieldGetFlags (
    FPD\_FormField formField
);
```

**Description**

Gets the field flags.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

**Return**

The field flags.

**Head file reference**

fpd\_docTempl.h: 4764

**FPDFFormFieldGetFont****Syntax**

```
FPD_Font FPDFFormFieldGetFont (
    FPD\_FormField formField
);
```

**Description**

Gets the font.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

**Return**

The PDF font.

**Head file reference**

fpd\_docTempl.h: 5371

## FPDFormFieldGetFontSize

### Syntax

```
FS_FLOAT FPDFormFieldGetFontSize (
    FPDFormField formField
);
```

### Description

Gets the font size.

### Parameter

---

formField	[In] The input PDF interactive form field.
-----------	--

---

### Return

The font size.

### Head file reference

fpd\_docTempl.h: 5371

## FPDFormFieldGetFullName

### Syntax

```
void FPDFormFieldGetFullName (
    FPDFormField formField,
    FS\_WideString* outName
);
```

### Description

Gets the full name of the field.

### Parameter

---

formField	[In] The input PDF interactive form field.
-----------	--

---

---

outName	[Out] The full name of the field.
---------	-----------------------------------

---

### Return

void

### Head file reference

fpd\_docTempl.h: 4745

## FPDFormFieldGetInterForm

### Syntax

```
FPD_InterForm FPDFormFieldGetInterForm (
    FPD\_FormField formField
);
```

### Description

Gets the interactive form which it belongs to.

### Parameter

---

formField	[In] The input PDF interactive form field.
-----------	--

---

### Return

The interactive form which it belongs to.

### Head file reference

fpd\_docTempl.h: 4773

## FPDFormFieldGetMappingName

### Syntax

```
void FPDFormFieldGetMappingName (
    FPD\_FormField formField,
    FS\_WideString* outName
);
```

### Description

Gets the mapping name to be used when exporting interactive form field data from the document.

### Parameter

---

formField	[In] The input PDF interactive form field.
-----------	--

---

---

outName	[Out] The mapping name
---------	------------------------

---

### Return

void

### Head file reference

fpd\_docTempl.h: 4890

## FPDFormFieldGetMaxLen

**Syntax**

```
FS_INT32 FPDFFormFieldGetMaxLen (
    FPDFormField formField
);
```

**Description**

Gets the maximum length of the field's text, in characters.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

**Return**

The maximum length of the field's text, in characters.

**Head file reference**

fpd\_docTempl.h: 5058

**FPDFFormFieldGetOptionLabel****Syntax**

```
void FPDFFormFieldGetOptionLabel (
    FPDFormField formField,
    FS_INT32 index,
    FS\_WideString* outLabel
);
```

**Description**

Gets the label of specified option.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

---

index	[In] The zero-based option index.
-------	-----------------------------------

---

---

outLabel	[Out] The label of the option.
----------	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5167

**FPDFFormFieldGetOptionValue**

**Syntax**

```
void FPDFFormFieldGetOptionValue (
    FPD\_FormField formField,
    FS\_INT32 index,
    FS\_WideString* outValue
);
```

**Description**

Gets the value of specified option.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

index	[In] The zero-based option index.
-------	-----------------------------------

---

outValue	[Out] The value of the option.
----------	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5178

**FPDFFormFieldGetRichTextString****Syntax**

```
void FPDFFormFieldGetRichTextString (
    FPD\_FormField formField,
    FS\_WideString* outText
);
```

**Description**

Gets the rich text string.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

outText	[Out] The rich text string.
---------	-----------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4959



## FPDFormFieldGetSelectedIndex

### Syntax

```
FS_INT32 FPDFormFieldGetSelectedIndex (
    FPDFormField formField,
    FS\_INT32 index
);
```

### Description

Gets the selected item index in the item array(include selected and unselected).

### Parameter

---

formField	[In] The input PDF interactive form field.
-----------	--

---

index	[In] The item index in the selected item array.
-------	---

---

### Return

The item index in all item array(include selected and unselected).

### Head file reference

fpd\_docTempl.h: 5086

## FPDFormFieldGetSelectedOptionIndex

### Syntax

```
FS_INT32 FPDFormFieldGetSelectedOptionIndex (
    FPDFormField formField,
    FS\_INT32 index
);
```

### Description

Gets the index of a selection option in all option array.

### Parameter

---

formField	[In] The input PDF interactive form field.
-----------	--

---

index	[In] zero-based index in "I" entry list.
-------	--

---

### Return

The index in all option array.

### Head file reference

fpd\_docTempl.h: 5329

## FPDFormFieldGetTopVisibleIndex

### Syntax

```
FS_INT32 FPDFormFieldGetTopVisibleIndex (
    FPDFormField formField
);
```

### Description

Gets the top visible option index.

### Parameter

---

formField	[In] The input PDF interactive form field.
-----------	--

---

### Return

The top visible option index.

### Head file reference

fpd\_docTempl.h: 5301

## FPDFormFieldGetType

### Syntax

```
FPD_FormFieldType FPDFormFieldGetType (
    FPDFormField formField
);
```

### Description

Gets the type of the field.

### Parameter

---

formField	[In] The input PDF interactive form field.
-----------	--

---

### Return

The type of the field.

### Head file reference

fpd\_docTempl.h: 4755

## FPDFormFieldGetValue

### Syntax

```
void FPDFormFieldGetValue (
```



```
FPD_FormField formField,  
FS_WideString* outValue  
);
```

**Description**

Gets the value of the field.

**Parameter**

---

formField	[In] The input PDF interactive form field.
outValue	[Out] The value of the field.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4989

**Note:**

- For a text field, the value is the text string;
- For a rich text field, the value is the rich text XML element;
- For a file field, the value is the path name of the file;
- For a radio button or check box field, the value is value string of the selected button;
- For a list box field, the value is the value of first selected item, if any;
- For a comb box field, the value is the text string.

**FPDFFormFieldInsertOption****Syntax**

```
FS_INT32 FPDFFormFieldInsertOption (  
    FPD_FormField formField,  
    FS_LPWSTR wszOptLabel,  
    FS_INT32 index,  
    FS_BOOL bNotify  
) ;
```

**Description**

Inserts an option at specified position.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---



wszOptLabel	[In] The label of the option.
-------------	-------------------------------

index	[In] The option index to insert at.
-------	-------------------------------------

bNotify	[In] Whether do notifying.
---------	----------------------------

**Return**

The inserted option index.

**Head file reference**

fpd\_docTempl.h: 5189

**FPDFFormFieldIsItemDefaultSelected****Syntax**

```
FS_BOOL FPDFFormFieldIsItemDefaultSelected (
    FPD\_FormField formField,
    FS\_INT32 index
);
```

**Description**

Checks whether the specified item's default selection flag is set.

**Parameter**

formField	[In] The input PDF interactive form field.
-----------	--

index	[In] The zero-based item index in the item array.
-------	---

**Return**

Non-zero means default selected, otherwise default unselected.

**Head file reference**

fpd\_docTempl.h: 5128

**FPDFFormFieldIsItemSelected****Syntax**

```
FS_BOOL FPDFFormFieldIsItemSelected (
    FPD\_FormField formField,
    FS\_INT32 index
);
```

**Description**

Checks whether the specified item has been selected.



**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

index	[In] The zero-based item index in the item array.
-------	---

---

**Return**

Non-zero means selected, otherwise unselected.

**Head file reference**

fpd\_docTempl.h: 5106

**FPDFormFieldIsOptionSelected****Syntax**

```
FS_BOOL FPDFormFieldIsOptionSelected (
    FPD\_FormField formField,
    FS\_INT32 iOptIndex
);
```

**Description**

Checks whether specified option has been selected.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

iOptIndex	[In] zero-based index of option in Opt list.
-----------	--

---

**Return**

Non-zero means selected, otherwise unselected.

**Head file reference**

fpd\_docTempl.h: 5339

**FPDFormFieldResetField****Syntax**

```
FS_BOOL FPDFormFieldResetField (
    FPD\_FormField formField,
    FS\_BOOL bNotify
);
```

**Description**

Resets the form field.



**Parameter**


---

formField	[In] The input PDF interactive form field.
bNotify	[In] Whether do notifying.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 4791

**FPDFormFieldSelectOption****Syntax**

```
FS_BOOL FPDFormFieldSelectOption (
    FPDFormField formField,
    FS\_INT32 iOptIndex,
    FS\_BOOL bSelected,
    FS\_BOOL bNotify
);
```

**Description**

Sets specified option to be selected.

**Parameter**


---

formField	[In] The input PDF interactive form field.
iOptIndex	[In] zero-based index of option in Opt list.
bSelected	[In] The input selection flag.
bNotify	[In] Whether do notifying.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 5349

**FPDFormFieldSetAdditionalAction****Syntax**

```
void FPDFormFieldSetAdditionalAction (
```



```
FPDFormField formField,  
FPD\_AAction aa  
);
```

**Description**

Sets the additional action of the field.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

aa	[In] The input additional action.
----	-----------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4849

**FPDFFormFieldSetAlternateName****Syntax**

```
void FPDFFormFieldSetAlternateName (  
    FPDFormField formField,  
    FS\_LPCSTR szName  
);
```

**Description**

Sets the alternate field name. ANSI version.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

szName	[In] The input alternate field name.
--------	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4870

**FPDFFormFieldSetAlternateNameW****Syntax**

```
void FPDFFormFieldSetAlternateNameW (
    FPD\_FormField formField,
    FS\_LPCWSTR wszName
);
```

**Description**

Sets the alternate field name. Unicode version.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

wszName	[In] The input alternate field name.
---------	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4880

**FPDFFormFieldSetDefaultStyle****Syntax**

```
void FPDFFormFieldSetDefaultStyle (
    FPD\_FormField formField,
    FS\_LPSTR szDefaultStyle
);
```

**Description**

Sets the default style string.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

szDefaultStyle	[In] The input default style string.
----------------	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4949

**FPDFFormFieldSetDefaultValue**

**Syntax**

```
FS_BOOL FPDFFormFieldSetDefaultValue (
    FPDFormField formField,
    FS\_LPCWSTR wszValue
);
```

**Description**

Sets the default value of the field.

**Parameter**

formField	[In] The input PDF interactive form field.
wszValue	[In] The input default field value.

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 5041

**Note:**

- For a text field, the value is the text string;
- For a rich text field, the value is the rich text XML element;
- For a file field, the value is the path name of the file;
- For a radio button or check box field, the value is value string of the selected button;
- For a list box field, the value is the value of first selected item, if any;
- For a comb box field, the value is the text string.

**FPDFFormFieldSetFieldFlags****Syntax**

```
void FPDFFormFieldSetFieldFlags (
    FPDFormField formField,
    FS\_DWORD dwFlags
);
```

**Description**

Sets the field flags.

**Parameter**

formField	[In] The input PDF interactive form field.
-----------	--



dwFlags	[In] The input field flags.
---------	-----------------------------

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4929

**FPDFormFieldSetItemDefaultSelection****Syntax**

```
void FPDFormFieldSetItemDefaultSelection (
    FPDFormField formField,
    FS\_INT32 index,
    FS\_BOOL bSelected
);
```

**Description**

Sets the default selection flag of specified item.

**Parameter**

formField	[In] The input PDF interactive form field.
-----------	--

index	[In] The zero-based item index in the item array.
-------	---

bSelected	[In] The input default selection flag.
-----------	--

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5138

**FPDFormFieldSetItemSelection****Syntax**

```
FS_BOOL FPDFormFieldSetItemSelection (
    FPDFormField formField,
    FS\_INT32 index,
    FS\_BOOL bSelected,
    FS\_BOOL bNotify
);
```

**Description**

Sets the selection flag of specified item.



**Parameter**


---

formField	[In] The input PDF interactive form field.
index	[In] The zero-based item index in the item array.
bSelected	[In] The input selection flag.
bNotify	[In] Whether do notifying.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 5116

**FPDFormFieldSetMappingName****Syntax**

```
void FPDFormFieldSetMappingName (
    FPDFormField formField,
    FS\_LPCSTR szName
);
```

**Description**

Sets the mapping name. ANSI version.

**Parameter**


---

formField	[In] The input PDF interactive form field.
szName	[In] The input mapping name.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4900

**FPDFormFieldSetMappingNameW****Syntax**

```
void FPDFormFieldSetMappingNameW (
    FPDFormField formField,
    FS\_LPCWSTR wszName
```



);

**Description**

Sets the mapping name. Unicode version.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

wszName	[In] The input mapping name.
---------	------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4910

**FPDFormFieldSetMaxLen****Syntax**

```
void FPDFormFieldSetMaxLen (
    FPD\_FormField formField,
    FS\_INT32 iMaxLen
);
```

**Description**

Sets the maximum length of the field's text, in characters.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

iMaxLen	[In] The input maximum length.
---------	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5067

**FPDFormFieldSetOptionLabel****Syntax**

```
FS_BOOL FPDFormFieldSetOptionLabel (
    FPD\_FormField formField,
```



```
FS_INT32 index,  
FS_LPCWSTR wszOptLabel,  
FS_BOOL bNotify  
);
```

**Description**

Sets the option label.

**Parameter**

formField	[In] The input PDF interactive form field.
index	[In] The zero-based index of the option.
wszOptLabel	[In] The new option label.
bNotify	[In] Whether do notifying.

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 5222

**FPDFormFieldSetOptionValue****Syntax**

```
FS_BOOL FPDFormFieldSetOptionValue (  
    FPD_FormField formField,  
    FS_INT32 index,  
    FS_LPCWSTR wszOptValue,  
    FS_BOOL bNotify  
);
```

**Description**

Sets the option value.

**Parameter**

formField	[In] The input PDF interactive form field.
index	[In] The zero-based index of the option.
wszOptValue	[In] The new option label.



---

bNotify	[In] Whether do notifying.
---------	----------------------------

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 5234

**FPDFormFieldSetRichTextString****Syntax**

```
void FPDFormFieldSetRichTextString (
    FPD_FormField formField,
    FS_LPCSTR szRichText
);
```

**Description**

Sets the rich text string. ANSI version.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

---

szRichText	[In] The input rich text string.
------------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4969

**FPDFormFieldSetRichTextStringW****Syntax**

```
void FPDFormFieldSetRichTextStringW (
    FPD_FormField formField,
    FS_LPCWSTR wszRichText
);
```

**Description**

Sets the rich text string. Unicode version.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

wszRichText	[In] The input rich text string.
-------------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4979

---

**FPDFormFieldSetTopVisibleIndex****Syntax**

```
void FPDFormFieldSetTopVisibleIndex (
    FPD\_FormField formField,
    FS\_INT32 index
);
```

**Description**

Sets the top visible option index.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

index	[In] The input option index.
-------	------------------------------

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5310

---

**FPDFormFieldSetValue****Syntax**

```
FS_BOOL FPDFormFieldSetValue (
    FPD\_FormField formField,
    FS\_LPCWSTR wszValue,
    FS\_BOOL bNotify
);
```

**Description**

Sets the value of the field. not applicable to non-unison radio box.

**Parameter**

---

formField	[In] The input PDF interactive form field.
-----------	--

---

wszValue	[In] The input field value.
bNotify	[In] Whether do notifying.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 5023

**Note:**

- For a text field, the value is the text string;
- For a rich text field, the value is the rich text XML element;
- For a file field, the value is the path name of the file;
- For a radio button or check box field, the value is value string of the selected button;
- For a list box field, the value is the value of first selected item, if any;
- For a comb box field, the value is the text string.

**FPDFormFieldUpdateUnisonStatus****Syntax**

```
void FPDFormFieldUpdateUnisonStatus (
    FPD\_FormField formField,
    FS\_BOOL bNotify
);
```

**Description**

Resets or recreate Opt array list and AP stream, for RadioButton only. This is used whenever unison status is changed.

**Parameter**


---

formField	[In] The input PDF interactive form field.
bNotify	[In] Whether do notifying.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5290

**FPD\_FormNotify**

## [Return from Used by](#)

### Description

A object representing a form notify handler. It is a event trigger for modifying form element. See [FPDInterFormGetFormNotify](#) , [FPDInterFormSetFormNotify](#) .

### Returned from

[FPDInterFormGetFormNotify](#)  
[FPDFFormNotifyNew](#)

### Used by

[FPDFFormNotifyDestroy](#)  
[FPDFFormNotifyNew](#)

## Callbacks

### Callbacks summary

#### [BeforeValueChange](#)

Triggered before a value changing.

#### [AfterValueChange](#)

Triggered after a value changed.

#### [BeforeSelectionChange](#)

Triggered before the selection changing.

#### [AfterSelectionChange](#)

Triggered after the selection changed.

#### [AfterCheckedStatusChange](#)

Triggered after the checked status changed.

#### [BeforeFormReset](#)

Triggered before form resetting.

#### [AfterFormReset](#)

Triggered after form reset.

#### [BeforeFormImportData](#)

Triggered before form importing data.

#### [AfterFormImportData](#)

Triggered after form data imported.

### Callbacks detail

#### BeforeValueChange

##### **Syntax**

```
typedef FS_INT32 (*BeforeValueChange)(  
    FS_LVOID clientData,  
    FPD_FormField field,  
    FS_WideString csValue  
)
```

**Description**

Triggered before a value changing.

**Parameter**

clientData	[In] The user-supplied data.
field	[In] The field which will be changed.
csValue	[In] The value to change to.

**Return**

Non-zero means event resolved.

**Head file reference**

fpd\_docExpT.h: 1099

**Group**

[FPD\\_FormNotifyCallbacksRec](#)

**AfterValueChange****Syntax**

```
typedef FS_INT32 (*AfterValueChange)(  
    FS_LPVVOID clientData,  
    FPD_FormField field  
)
```

**Description**

Triggered after a value changed.

**Parameter**

clientData	[In] The user-supplied data.
field	[In] The field which was just changed.

**Return**

Non-zero means event resolved.

**Head file reference**

fpd\_docExpT.h: 1110

**Group**

[FPD\\_FormNotifyCallbacksRec](#)



## BeforeSelectionChange

### Syntax

```
typedef FS_INT32 (*BeforeSelectionChange)(  
    FS_LPVVOID clientData,  
    FPD_FormField field,  
    FS_WideString csValue  
)
```

### Description

Triggered before the selection changing.

### Parameter

clientData	[In] The user-supplied data.
field	[In] The field who's selection will be changed.
csValue	[In] The option label to change to.

### Return

Non-zero means event resolved.

### Head file reference

fpd\_docExpT.h: 1123

### Group

[FPD\\_FormNotifyCallbacksRec](#)

## AfterSelectionChange

### Syntax

```
typedef FS_INT32 (*AfterSelectionChange)(  
    FS_LPVVOID clientData,  
    FPD_FormField field  
)
```

### Description

Triggered after the selection changed.

### Parameter

clientData	[In] The user-supplied data.
field	[In] The field who's selection was just changed.



**Return**

Non-zero means event resolved.

**Head file reference**

fpd\_docExpT.h: 1135

**Group**

[FPD\\_FormNotifyCallbacksRec](#)

AfterCheckedStatusChange

**Syntax**

```
typedef FS_INT32 (*AfterCheckedStatusChange)(  
    FS_LPVVOID clientData,  
    FPDFormField field,  
    FS_ByteArray statusArray  
)
```

**Description**

Triggered after the checked status changed.

**Parameter**

clientData	[In] The user-supplied data.
field	[In] The field who's checked status was just changed.
statusArray	[In] The changed status array.

**Return**

Non-zero means event resolved.

**Head file reference**

fpd\_docExpT.h: 1148

**Group**

[FPD\\_FormNotifyCallbacksRec](#)

BeforeFormReset

**Syntax**

```
typedef FS_INT32 (*BeforeFormReset)(  
    FS_LPVVOID clientData,  
    FPD_InterForm form  
)
```



**Description**

Triggered before form resetting.

**Parameter**

clientData	[In] The user-supplied data.
form	[In] The interactive form to reset.

**Return**

Non-zero means event resolved.

**Head file reference**

fpd\_docExpT.h: 1160

**Group**

[FPD\\_FormNotifyCallbacksRec](#)

**AfterFormReset****Syntax**

```
typedef FS_INT32 (*AfterFormReset)(  
    FS_LPVVOID clientData,  
    FPD_InterForm form  
)
```

**Description**

Triggered after form reset.

**Parameter**

clientData	[In] The user-supplied data.
form	[In] The interactive form who has just reset.

**Return**

Non-zero means event resolved.

**Head file reference**

fpd\_docExpT.h: 1172

**Group**

[FPD\\_FormNotifyCallbacksRec](#)

**BeforeFormImportData**

**Syntax**

```
typedef FS_INT32 (*BeforeFormImportData)(  
    FS\_LPVVOID clientData,  
    FPD\_InterForm form  
) ;
```

**Description**

Triggered before form importing data.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

form	[In] The interactive form to import data.
------	---

---

**Return**

Non-zero means event resolved.

**Head file reference**

[fpd\\_docExpT.h: 1184](#)

**Group**

[FPD\\_FormNotifyCallbacksRec](#)

**AfterFormImportData****Syntax**

```
typedef FS_INT32 (*AfterFormImportData)(  
    FS\_LPVVOID clientData,  
    FPD\_InterForm form  
) ;
```

**Description**

Triggered after form data imported.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

form	[In] The interactive form who has just imported data.
------	---

---

**Return**

Non-zero means event resolved.

**Head file reference**

[fpd\\_docExpT.h: 1196](#)



**Group**[FPD\\_FormNotifyCallbacksRec](#)

## Functions

### Functions summary

[\*\*FPDFFormNotifyDestroy\*\*](#)

Destroys a form notify object.

[\*\*FPDFFormNotifyNew\*\*](#)

Creates a form notify object.

### Functions detail

[\*\*FPDFFormNotifyDestroy\*\*](#)**Syntax**

```
void FPDFFormNotifyDestroy (
    FPD\_FormNotify formNotify
);
```

**Description**

Destroys a form notify object.

**Parameter**

---

formNotify	[In] The <a href="#"><u>FPD_FormNotify</u></a> object to be destroyed.
------------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3835

[\*\*FPDFFormNotifyNew\*\*](#)**Syntax**

```
FPD_FormNotify FPDFFormNotifyNew (
    FPD\_FormNotifyCallbacks callbacks
);
```

**Description**

Creates a form notify object.

**Parameter**

---

callbacks	[In] The callback set.
-----------	------------------------

---

**Return**

A newly form notify object.

**Head file reference**

fpd\_docTempl.h: 3831

**Related method**

[FPDInterFormGetFormNotify](#)  
[FPDInterFormSetFormNotify](#)  
[FPDFFormNotifyDestroy](#)

# FPD\_FormObject

## Description

This object corresponds to a PDF form XObject ( *see Section 4.9, Form XObjects, in the PDF Reference* ).

## Functions

### Functions summary

[\*\*FPDFFormObjectCalcBoundingBox\*\*](#)

Calculates the bounding box.

[\*\*FPDFFormObjectDestroy\*\*](#)

Destroys the PDF form object.

[\*\*FPDFFormObjectGetForm\*\*](#)

Gets the form.

[\*\*FPDFFormObjectGetTransformMatrix\*\*](#)

Gets the transformation matrix.

[\*\*FPDFFormObjectNew\*\*](#)

Creates a new empty PDF form object.

[\*\*FPDFFormObjectSetForm\*\*](#)

Sets the form.

[\*\*FPDFFormObjectSetTransformMatrix\*\*](#)

Sets the transformation matrix.

[\*\*FPDFFormObjectTransform\*\*](#)

Transforms the path object.

### Functions detail

**FPDFFormObjectCalcBoundingBox**

#### Syntax

```
void FPDFFormObjectCalcBoundingBox (
    FPD PageObject objForm
);
```

**Description**

Calculates the bounding box.

**Parameter**

---

objForm	[In] The input PDF form object.
---------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1828

**FPDFormObjectDestroy****Syntax**

```
void FPDFormObjectDestroy (
    FPD\_PageObject objForm
);
```

**Description**

Destroys the PDF form object.

**Parameter**

---

objForm	[In] The input PDF form object.
---------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1809

**FPDFormObjectGetForm****Syntax**

```
FPD_Form FPDFormObjectGetForm (
    FPD\_PageObject objForm
);
```

**Description**

Gets the form.

**Parameter**

---

objForm	[In] The input PDF form object.
---------	---------------------------------

---

**Return**

The form.

**Head file reference**

fpd\_pageobjTempl.h: 1857

**FPDFormObjectGetTransformMatrix****Syntax**

```
void FPDFormObjectGetTransformMatrix (
    FPD\_PageObject objForm,
    FS\_AffineMatrix* outmatrix
);
```

**Description**

Gets the transformation matrix.

**Parameter**

---

objForm	[In] The input PDF form object.
---------	---------------------------------

---

outmatrix	[Out] It receives the transformation matrix.
-----------	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1837

**FPDFormObjectNew****Syntax**

```
FPD_PageObject FPDFormObjectNew (void );
```

**Description**

Creates a new empty PDF form object.

**Return**

A new empty PDF form object.

**Head file reference**

fpd\_pageobjTempl.h: 1800

**FPDFormObjectSetForm**

**Syntax**

```
void FPDFFormObjectSetForm (  
    FPD\_PageObject objForm,  
    FPD\_Form form  
) ;
```

**Description**

Sets the form.

**Parameter**

---

objForm	[In] The input PDF form object.
---------	---------------------------------

---

form	[In] The input form.
------	----------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1866

**FPDFFormObjectSetTransformMatrix****Syntax**

```
void FPDFFormObjectSetTransformMatrix (  
    FPD\_PageObject objForm,  
    const FS\_AffineMatrix\* matrix  
) ;
```

**Description**

Sets the transformation matrix.

**Parameter**

---

objForm	[In] The input PDF form object.
---------	---------------------------------

---

matrix	[In] The transformation matrix.
--------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1847

**FPDFFormObjectTransform**

**Syntax**

```
void FPDFFormObjectTransform (
    FPD\_PageObject objForm,
    FS\_AffineMatrix matrix
);
```

**Description**

Transforms the path object.

**Parameter**

---

objForm	[In] The input PDF form object.
---------	---------------------------------

---

matrix	[In] The transformation matrix.
--------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1818

## FPD\_FT\_Face

**[Return from](#)****Description**

FreeType root face class structure. A face object models a typeface in a font file.

**Returned from****[FPDFFontGetFace](#)**

## FPD\_Function

**[Return from Used by](#)****Description**

No document exists.

**Returned from****[FPDSHadingPatternGetFunc](#)****Used by**

[FPDMeshStreamNew](#)

## FPD\_FXFont

[Return from Used by](#)

### Description

The [FPD\\_FXFont](#) object is used to manipulate the Foxit GE font. See [FPDFFontGetFXFont](#).

### Returned from

[FPDFFontGetFXFont](#)  
[FPDFXFontEncodingNew](#)

### Used by

[FPDDocAddFXFont](#)  
[FPDFFontFXFontGetFaceName](#)  
[FPDFFontFXFontGetFamilyName](#)  
[FPDFFontFXFontGetPsName](#)  
[FPDFFontFXFontIsBold](#)  
[FPDFFontFXFontIsItalic](#)  
[FPDFFontFXFontLoadSubst](#)  
[FPDFXFontEncodingCharCodeFromUnicode](#)  
[FPDFXFontEncodingDestroy](#)  
[FPDFXFontEncodingGlyphFromCharCode](#)  
[FPDFXFontEncodingCharCodeEx](#)  
[FPDFXFontEncodingIsUnicodeCompatible](#)  
[FPDFXFontEncodingUnicodeFromCharCode](#)

## FPD\_FXFontEncoding

[Return from Used by](#)

### Description

It works with a font to translate character codes into glyph indices in that font. It also deals with Unicode mapping (if supported). See [FPDFXFontEncodingNew](#), [FPDFXFontEncodingDestroy](#).

### Returned from

[FPDFXFontEncodingNew](#)

### Used by

[FPDFXFontEncodingCharCodeFromUnicode](#)  
[FPDFXFontEncodingDestroy](#)  
[FPDFXFontEncodingGlyphFromCharCode](#)  
[FPDFXFontEncodingCharCodeEx](#)

[\*\*FPDFXFontEncodingIsUnicodeCompatible\*\*](#)  
[\*\*FPDFXFontEncodingUnicodeFromCharCode\*\*](#)

## Definitions

### Definitions summary

#### [\*\*FPD\\_FXENCODING\\_INTERNAL\*\*](#)

Whatever internal encoding in the font.

#### [\*\*FPD\\_FXENCODING\\_UNICODE\*\*](#)

Unicode encoding.

### Definitions detail

#### **FPD\_FXENCODING\_INTERNAL**

##### **Syntax**

```
#define FPD_FXENCODING_INTERNAL 0
```

##### **Description**

Whatever internal encoding in the font.

##### **Group**

[FPDFXEncodingType](#)

##### **Head file reference**

fpd\_resourceExpT.h: 373

#### **FPD\_FXENCODING\_UNICODE**

##### **Syntax**

```
#define FPD_FXENCODING_UNICODE 1
```

##### **Description**

Unicode encoding.

##### **Group**

[FPDFXEncodingType](#)

##### **Head file reference**

fpd\_resourceExpT.h: 375

## Functions

### Functions summary

#### [\*\*FPDFXFontEncodingCharCodeFromUnicode\*\*](#)

Gets a charcode from a Unicode. Return -1 for unknown.

**FPDFXFontEncodingDestroy**

Destroys the font encoding.

**FPDFXFontEncodingGlyphFromCharCode**

Gets the glyph index for a charcode. Return -1 for unknown. For embedded font only.

**FPDFXFontEncodingGlyphFromCharCodeEx**

Gets the glyph index for a charcode. Return -1 for unknown. For embedded font only.

**FPDFXFontEncodingIsUnicodeCompatible**

Checks whether the encoding is Unicode compatible.

**FPDFXFontEncodingNew**

It works with a font to translate character codes into glyph indices in that font. It also deals with Unicode mapping (if supported).

**FPDFXFontEncodingUnicodeFromCharCode**

Gets a unicode string for a charcode. Return empty for unknown.

**Functions detail****FPDFXFontEncodingCharCodeFromUnicode****Syntax**

```
FS_DWORD FPDFXFontEncodingCharCodeFromUnicode (
    FPD\_FXFontEncoding fontEncoding,
    FS\_WCHAR Unicode
);
```

**Description**

Gets a charcode from a Unicode. Return -1 for unknown.

**Parameter**

fontEncoding	[In] The input font encoding.
--------------	-------------------------------

Unicode	[In] Input a unicode.
---------	-----------------------

**Return**

The charcode for the unicode.

**Head file reference**

fpd\_resourceTempl.h: 2228

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FPDFXFontEncodingDestroy****Syntax**

```
void FPDFXFontEncodingDestroy (
    FPD\_FXFontEncoding fontEncoding
);
```



**Description**

Destroys the font encoding.

**Parameter**

---

fontEncoding	[In] The input font encoding.
--------------	-------------------------------

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 2195

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDFXFontEncodingGlyphFromCharCode****Syntax**

```
FS_DWORD FPDFXFontEncodingGlyphFromCharCode (
    FPD\_FXFontEncoding fontEncoding,
    FS\_DWORD charcode
);
```

**Description**

Gets the glyph index for a charcode. Return -1 for unknown. For embedded font only.

**Parameter**

---

fontEncoding	[In] The input font encoding.
--------------	-------------------------------

---

---

charcode	[In] Input a charcode.
----------	------------------------

---

**Return**

The glyph index of the charcode.

**Head file reference**

fpd\_resourceTempl.h: 2205

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDFXFontEncodingGlyphFromCharCodeEx****Syntax**

```
FS_DWORD FPDFXFontEncodingGlyphFromCharCodeEx (
```



---

```
FPD_FXFontEncoding fontEncoding,
FS_DWORD charcode,
FS_INT32 encodingType
);
```

**Description**

Gets the glyph index for a charcode. Return -1 for unknown. For embedded font only.

**Parameter**

fontEncoding	[In] The input font encoding.
charcode	[In] Input a charcode.
encodingType	[In] The input encoding type. See <a href="#">FPDFXEncodingType</a> .

**Return**

The glyph index of the charcode.

**Head file reference**

fpd\_resourceTempl.h: 2249

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDFXFontEncodingIsUnicodeCompatible****Syntax**

```
FS_BOOL FPDFXFontEncodingIsUnicodeCompatible (
    FPD_FXFontEncoding fontEncoding
);
```

**Description**

Checks whether the encoding is Unicode compatible.

**Parameter**

fontEncoding	[In] The input font encoding.
--------------	-------------------------------

**Return**

Checks whether the encoding is Unicode compatible.

**Head file reference**

fpd\_resourceTempl.h: 2239

**Since**

## [SDK\\_LATEEST\\_VERSION > 1.0](#)

### FPDFXFontEncodingNew

#### Syntax

```
FPD_FXFontEncoding FPDFXFontEncodingNew (
    FPD\_Font fpdFont
);
```

#### Description

It works with a font to translate character codes into glyph indices in that font. It also deals with Unicode mapping (if supported).

#### Parameter

---

fpdFont	[In] The input PDF font object.
---------	---------------------------------

---

#### Return

A new font encoding.

#### Head file reference

fpd\_resourceTempl.h: 2185

#### Since

## [SDK\\_LATEEST\\_VERSION > 1.0](#)

### FPDFXFontEncodingUnicodeFromCharCode

#### Syntax

```
void FPDFXFontEncodingUnicodeFromCharCode (
    FPD\_FXFontEncoding fontEncoding,
    FS\_DWORD charcode,
    FS\_WideString* outUnicodeString
);
```

#### Description

Gets a unicode string for a charcode. Return empty for unknown.

#### Parameter

---

fontEncoding	[In] The input font encoding.
--------------	-------------------------------

---

---

charcode	[In] Input a charcode.
----------	------------------------

---

---

outUnicodeString	[Out] It receives the unicode string for the charcode.
------------------	--

---

**Return**

void.

**Head file reference**

fpd\_resourceTempl.h: 2216

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FPD\_FxgeDevice

### Description

Foxit own FXGE rendering device, using the Foxit FXGE device driver. Supporting output to following formats:

- 8bppMask: the output will be the alpha channel.
- 8bppRgb without palette: the output will be grayscale.
- Rgb, Rgb32: normal RGB output.
- Argb: RGB with alpha channel output.

### Functions

#### Functions summary

[\*\*FPDFxgeDeviceAttach\*\*](#)

Attach a bitmap.

[\*\*FPDFxgeDeviceCreate\*\*](#)

Creates a new bitmap and attach to this device.

[\*\*FPDFxgeDeviceDestroy\*\*](#)

Destroys the GE rendering device object.

[\*\*FPDFxgeDeviceNew\*\*](#)

Creates a new empty GE rendering device object.

#### Functions detail

##### FPDFxgeDeviceAttach

**Syntax**

```
void FPDFxgeDeviceAttach (
    FPD\_RenderDevice dc,
    FS\_DIBitmap bitmap,
    FS\_INT32 ditherBits
);
```

**Description**

Attach a bitmap.

**Parameter**

---

dc	[In] The input GE rendering device object.
----	--

---

bitmap	[In] The input bitmap to be attached.
--------	---------------------------------------

---

ditherBits	[In] The input dither bits.
------------	-----------------------------

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 845

**FPDFxgeDeviceCreate****Syntax**

```
void FPDFxgeDeviceCreate (
    FPD\_RenderDevice dc,
    FS\_INT32 width,
    FS\_INT32 height,
    FS\_DIB\_Format format,
    FS\_INT32 ditherBits
);
```

**Description**

Creates a new bitmap and attach to this device.

**Parameter**


---

dc	[In] The input GE rendering device object.
----	--

---

width	[In] The bitmap width.
-------	------------------------

---

height	[In] The bitmap height
--------	------------------------

---

format	[In] The bitmap format.
--------	-------------------------

---

ditherBits	[In] The optional dithering bits. 0 for no dithering
------------	--

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 823

**Note:**The bitmap will be destroyed when the device destructs.

## FPDFxgeDeviceDestroy

### Syntax

```
void FPDFxgeDeviceDestroy (
    FPD\_RenderDevice dc
);
```

### Description

Destroys the GE rendering device object.

### Parameter

---

dc	[In] The input GE rendering device object.
----	--

---

### Return

void

### Head file reference

[fpd\\_renderTempl.h](#): 836

## FPDFxgeDeviceNew

### Syntax

```
FPD_RenderDevice FPDFxgeDeviceNew (void );
```

### Description

Creates a new empty GE rendering device object.

### Return

A new empty GE rendering device object.

### Head file reference

[fpd\\_renderTempl.h](#): 815

## FPD\_GeneralState

### [Return from Used by](#)

### Description

The general state for page rendering. See [FPDGeneralStateNew](#) , [FPDGeneralStateDestroy](#) .

## Returned from

[\*\*FPDPageObjectGetGeneralState\*\*](#)  
[\*\*FPDGeneralStateNew\*\*](#)

## Used by

[\*\*FPDPageArchiveLoaderLoadGeneralState\*\*](#)  
[\*\*FPDPageArchiveSaverSaveGeneralState\*\*](#)  
[\*\*FPDPageObjectSetGeneralState\*\*](#)  
[\*\*FPDGeneralStateDestroy\*\*](#)  
[\*\*FPDGeneralStateGetAlpha\*\*](#)  
[\*\*FPDGeneralStateGetBlendType\*\*](#)  
[\*\*FPDGeneralStateGetModify\*\*](#)  
[\*\*FPDGeneralStateIsNull\*\*](#)  
[\*\*FPDGeneralStateSetBlendMode\*\*](#)  
[\*\*FPDGeneralStateSetBlendType\*\*](#)  
[\*\*FPDGeneralStateSetFillAlpha\*\*](#)  
[\*\*FPDGeneralStateSetRenderIntent\*\*](#)  
[\*\*FPDGeneralStateSetSoftMask\*\*](#)  
[\*\*FPDGeneralStateSetSoftMaskMatrix\*\*](#)  
[\*\*FPDGeneralStateSetStrokeAlpha\*\*](#)

## Functions

### Functions summary

#### [\*\*FPDGeneralStateDestroy\*\*](#)

Destroys the PDF general state data object.

#### [\*\*FPDGeneralStateGetAlpha\*\*](#)

Gets the current filling or stroking alpha constant.

#### [\*\*FPDGeneralStateGetBlendType\*\*](#)

Gets the blend mode.

#### [\*\*FPDGeneralStateGetModify\*\*](#)

The interface helps init the object if the object is NULL.

#### [\*\*FPDGeneralStateIsNull\*\*](#)

Tests whether the general state object is [NULL](#) or not.

#### [\*\*FPDGeneralStateNew\*\*](#)

Creates a new empty PDF general state object.

#### [\*\*FPDGeneralStateSetBlendMode\*\*](#)

Sets the current blend mode name.

#### [\*\*FPDGeneralStateSetBlendType\*\*](#)

Sets the current blend mode to be used in the transparent imaging model.

#### [\*\*FPDGeneralStateSetFillAlpha\*\*](#)

Same as stroking alpha, but for non-stroking operations.

#### [\*\*FPDGeneralStateSetRenderIntent\*\*](#)

Sets the rendering intent.

#### [\*\*FPDGeneralStateSetSoftMask\*\*](#)

Sets the current soft mask, specifying the mask shape or mask opacity values to be used in the transparent imaging model.

#### [\*\*FPDGeneralStateSetSoftMaskMatrix\*\*](#)

Sets the matrix of the current soft mask.

**[FPDGeneralStateSetStrokeAlpha](#)**

Sets The current stroking alpha constant, specifying the constant shape or constant opacity value to be used for stroking operations in the transparent imaging model.

**Functions detail****FPDGeneralStateDestroy****Syntax**

```
void FPDGeneralStateDestroy (
    FPD\_GeneralState genState
);
```

**Description**

Destroys the PDF general state data object.

**Parameter**

---

genState	[In] The input PDF general state data object.
----------	---

---

**Return**

void

**Head file reference**

[fpd\\_pageobjTempl.h](#): 705

**FPDGeneralStateGetAlpha****Syntax**

```
FS_INT32 FPDGeneralStateGetAlpha (
    FPD\_GeneralState genState,
    FS\_BOOL bStroke
);
```

**Description**

Gets the current filling or stroking alpha constant.

**Parameter**

---

genState	[In] The input PDF general state data object.
----------	---

---

bStroke	[In] Whether to get the current stroking alpha constant.
---------	--

---

**Return**

The current filling or stroking alpha constant.

**Head file reference**

fpd\_pageobjTempl.h: 733

## FPDGeneralStateGetBlendType

### Syntax

```
FS_INT32 FPDGeneralStateGetBlendType (
    FPD\_GeneralState genState
);
```

### Description

Gets the blend mode.

### Parameter

---

genState	[In] The input PDF general state data object.
----------	---

---

### Return

The blend mode.

### Head file reference

fpd\_pageobjTempl.h: 724

## FPDGeneralStateGetModify

### Syntax

```
void FPDGeneralStateGetModify (
    FPD\_GeneralState genState
);
```

### Description

The interface helps init the object if the object is NULL.

### Parameter

---

genState	[In] The input PDF general state object.
----------	--

---

### Return

void.

### Head file reference

fpd\_pageobjTempl.h: 813

### Since

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FPDGeneralStateIsNull



**Syntax**

```
FS_BOOL FPDGeneralStateIsNull (
    FPD GeneralState genState
);
```

**Description**

Tests whether the general state object is [NULL](#) or not.

**Parameter**

---

genState	[In] The input PDF general state data object.
----------	---

---

**Return**

Non-zero means [NULL](#) , otherwise not [NULL](#) .

**Head file reference**

fpd\_pageobjTempl.h: 804

**FPDGeneralStateNew****Syntax**

```
FPD_GeneralState FPDGeneralStateNew (void );
```

**Description**

Creates a new empty PDF general state object.

**Return**

A new empty PDF general state object.

**Head file reference**

fpd\_pageobjTempl.h: 696

**FPDGeneralStateSetBlendMode****Syntax**

```
void FPDGeneralStateSetBlendMode (
    FPD GeneralState genState,
    BlendMode
);
```

**Description**

Sets the current blend mode name.

**Parameter**

---

genState	[In] The input PDF general state data object.
----------	---

---

BlendMode	[In] The input current blend mode name.
-----------	---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 743

**FPDGeneralStateSetBlendType****Syntax**

```
void FPDGeneralStateSetBlendType (
    FPD\_GeneralState genState,
    FS\_INT32 iBlendType
);
```

**Description**

Sets the current blend mode to be used in the transparent imaging model.

**Parameter**

---

genState	[In] The input PDF general state data object.
----------	---

---

iBlendType	[In] The input blend type.
------------	----------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 753

**FPDGeneralStateSetFillAlpha****Syntax**

```
void FPDGeneralStateSetFillAlpha (
    FPD\_GeneralState genState,
    FS\_FLOAT fFillAlpha
);
```

**Description**

Same as stroking alpha, but for non-stroking operations.

**Parameter**

---

genState	[In] The input PDF general state data object.
----------	---

---

---

ffFillAlpha	[In] The input current filling alpha constant.
-------------	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 794

**FPDGeneralStateSetRenderIntent****Syntax**

```
void FPDGeneralStateSetRenderIntent (
    FPD\_GeneralState genState,
    FS LPCSTR ri
);
```

**Description**

Sets the rendering intent.

**Parameter**

---

genState	[In] The input PDF general state data object.
----------	---

---

---

ri	[In] The input rendering intent.
----	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 714

**FPDGeneralStateSetSoftMask****Syntax**

```
void FPDGeneralStateSetSoftMask (
    FPD\_GeneralState genState,
    FPD\_Object softMask
);
```

**Description**

Sets the current soft mask, specifying the mask shape or mask opacity values to be used in the transparent imaging model.

**Parameter**

---

genState	[In] The input PDF general state data object.
----------	---

---

---

softMask	[In] The input current soft mask.
----------	-----------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 763

**FPDGeneralStateSetSoftMaskMatrix****Syntax**

```
void FPDGeneralStateSetSoftMaskMatrix (
    FPD\_GeneralState genState,
    SoftMaskMatrix
);
```

**Description**

Sets the matrix of the current soft mask.

**Parameter**

---

genState	[In] The input PDF general state data object.
----------	---

---

---

SoftMaskMatrix	[In] The input matrix of the current soft mask.
----------------	---

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 773

**FPDGeneralStateSetStrokeAlpha****Syntax**

```
void FPDGeneralStateSetStrokeAlpha (
    FPD\_GeneralState genState,
    FS\_FLOAT fStrokeAlpha
);
```

**Description**

Sets The current stroking alpha constant, specifying the constant shape or constant opacity value to be used for stroking operations in the transparent imaging model.

**Parameter**

---

genState	[In] The input PDF general state data object.
----------	---

---



---

fStrokeAlpha	[In] The input current stroking alpha constant.
--------------	---

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 783

## FPD\_GraphState

**Return from Used by**

### Description

The graph state for graphic rendering.

### Returned from

[FPDPageObjectGetGraphState](#)  
[FPDGraphStateNew](#)

### Used by

[FPDPageArchiveLoaderLoadGraphState](#)  
[FPDPageArchiveSaverSaveGraphState](#)  
[FPDPageObjectSetGraphState](#)  
[FPDPathObjectSetGraphState](#)  
[FPDRenderDeviceDrawPath](#)  
[FPDGraphStateDestroy](#)  
[FPDGraphStateGetDashArray](#)  
[FPDGraphStateGetDashCount](#)  
[FPDGraphStateGetDashPhase](#)  
[FPDGraphStateGetLineCap](#)  
[FPDGraphStateGetLineJoin](#)  
[FPDGraphStateGetLineWidth](#)  
[FPDGraphStateGetMiterLimit](#)  
[FPDGraphStateGetModify](#)  
[FPDGraphStateIsNull](#)  
[FPDGraphStateSetDashArray](#)  
[FPDGraphStateSetDashCount](#)  
[FPDGraphStateSetDashPhase](#)  
[FPDGraphStateSetLineCap](#)  
[FPDGraphStateSetLineJoin](#)  
[FPDGraphStateSetLineWidth](#)  
[FPDGraphStateSetMiterLimit](#)

### Enumerations

#### Enumerations summary

[FPD\\_LineCap](#)

Line cap style enumeration.

#### **FPD\_LineCap**

Line cap style enumeration.

### Enumerations detail

#### **FPD\_LineCap**

##### **Syntax**

```
enum FPD_LineCap{  
    FPD\_LineCapButt,  
    FPD\_LineCapRound,  
    FPD\_LineCapSquare  
};
```

##### **Description**

Line cap style enumeration.

##### **Head file reference**

fpd\_pageobjExpT.h: 94

#### **[FPD\\_LineCapButt](#)**

Butt cap. The stroke is squared off at the endpoint of the path.

#### **[FPD\\_LineCapRound](#)**

Round cap. A semicircular arc with a diameter equal to the line width is drawn around the endpoint and filled in.

#### **[FPD\\_LineCapSquare](#)**

Projecting square cap. The stroke continues beyond the endpoint of the path for a distance equal to half the line width and is squared off.

#### **FPD\_LineJoin**

##### **Syntax**

```
enum FPD_LineJoin{  
    FPD\_LineJoinMiter,  
    FPD\_LineJoinRound,  
    FPD\_LineJoinBevel  
};
```

##### **Description**

Line join style enumeration.

##### **Head file reference**

fpd\_pageobjExpT.h: 106

#### **[FPD\\_LineJoinMiter](#)**

Miter join. The outer edges of the strokes for the two segments are extended until they meet at an angle, as in a picture frame.

#### **[FPD\\_LineJoinRound](#)**

Round join. An arc of a circle with a diameter equal to the line width is drawn around the point where the two segments meet, connecting the outer edges of the strokes for the two segments.

#### **FPD\_LineJoinBevel**

Bevel join. The two segments are finished with butt caps and the resulting notch beyond the ends of the segments is filled with a triangle.

## Functions

### Functions summary

#### **[FPDGraphStateDestroy](#)**

Destroys a graphic state object.

#### **[FPDGraphStateGetDashArray](#)**

Gets the dash array.

#### **[FPDGraphStateGetDashCount](#)**

Gets the total size of dash array.

#### **[FPDGraphStateGetDashPhase](#)**

Gets the dash phase for line dash pattern.

#### **[FPDGraphStateGetLineCap](#)**

Gets the line cap style.

#### **[FPDGraphStateGetLineJoin](#)**

Gets the line join style.

#### **[FPDGraphStateGetLineWidth](#)**

Gets the line width.

#### **[FPDGraphStateGetMiterLimit](#)**

Gets the miter limit for line join.

#### **[FPDGraphStateGetModify](#)**

The interface helps init the object if the object is NULL.

#### **[FPDGraphStateIsNull](#)**

Tests whether the graphic state object is [NULL](#) or not.

#### **[FPDGraphStateNew](#)**

Creates a new empty PDF graphic state object.

#### **[FPDGraphStateSetDashArray](#)**

Sets the dash phase for line dash pattern.

#### **[FPDGraphStateSetDashCount](#)**

In order to keep heap integrity, the function is used to allocate enough buffer for dash array.

#### **[FPDGraphStateSetDashPhase](#)**

Sets the new dash phase for line dash pattern.

#### **[FPDGraphStateSetLineCap](#)**

Sets the new line cap style.

#### **[FPDGraphStateSetLineJoin](#)**

Sets the new style of line join.

#### **[FPDGraphStateSetLineWidth](#)**

Sets new width of lines.

#### **[FPDGraphStateSetMiterLimit](#)**

Sets the new miter limit for line join.

## Functions detail

### FPDGraphStateDestroy

#### Syntax

```
void FPDGraphStateDestroy (
    FPD\_GraphState graphState
);
```

#### Description

Destroys a graphic state object.

#### Parameter

---

graphState	[In] The graphic state object to be destroyed.
------------	--

---

#### Return

void

#### Head file reference

fpd\_pageobjTempl.h: 840

### FPDGraphStateGetDashArray

#### Syntax

```
FS_FLOAT* FPDGraphStateGetDashArray (
    FPD\_GraphState graphState
);
```

#### Description

Gets the dash array.

#### Parameter

---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

#### Return

The dash array used by graphic state object.

#### Head file reference

fpd\_pageobjTempl.h: 868

### FPDGraphStateGetDashCount

#### Syntax

```
FS_INT32 FPDGraphStateGetDashCount (
    FPD\_GraphState graphState
);
```

**Description**

Gets the total size of dash array.

**Parameter**

---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

**Return**

The total size of dash array.

**Head file reference**

fpd\_pageobjTempl.h: 859

**FPDGraphStateGetDashPhase****Syntax**

```
FS_FLOAT FPDGraphStateGetDashPhase (
    FPD\_GraphState graphState
);
```

**Description**

Gets the dash phase for line dash pattern.

**Parameter**

---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

**Return**

The dash phase for line dash pattern.

**Head file reference**

fpd\_pageobjTempl.h: 877

**FPDGraphStateGetLineCap****Syntax**

```
FPD_LineCap FPDGraphStateGetLineCap (
    FPD\_GraphState graphState
);
```

**Description**

Gets the line cap style.

**Parameter**

---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

**Return**

The line cap style.

**Head file reference**

fpd\_pageobjTempl.h: 896

**FPDGraphStateGetLineJoin****Syntax**

```
FPD_LineJoin FPDGraphStateGetLineJoin (
    FPD\_GraphState graphState
);
```

**Description**

Gets the line join style.

**Parameter**

---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

**Return**

The line join style.

**Head file reference**

fpd\_pageobjTempl.h: 915

**FPDGraphStateGetLineWidth****Syntax**

```
FS_FLOAT FPDGraphStateGetLineWidth (
    FPD\_GraphState graphState
);
```

**Description**

Gets the line width.

**Parameter**

---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

**Return**

The line width.

**Head file reference**

fpd\_pageobjTempl.h: 953

## FPDGraphStateGetMiterLimit

### Syntax

```
FS_FLOAT FPDGraphStateGetMiterLimit (
    FPD\_GraphState graphState
);
```

### Description

Gets the miter limit for line join.

### Parameter

---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

### Return

The miter limit for line join.

### Head file reference

fpd\_pageobjTempl.h: 934

## FPDGraphStateGetModify

### Syntax

```
void FPDGraphStateGetModify (
    FPD\_GraphState graphState
);
```

### Description

The interface helps init the object if the object is NULL.

### Parameter

---

graphState	[In] The input PDF graph state object.
------------	--

---

### Return

void.

### Head file reference

fpd\_pageobjTempl.h: 981

### Since

[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FPDGraphStateIsNull

**Syntax**

```
FS_BOOL FPDGraphStateIsNull (
    FPD\_GraphState graphState
);
```

**Description**

Tests whether the graphic state object is [NULL](#) or not.

**Parameter**

---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

**Return**

Non-zero means [NULL](#) , otherwise not [NULL](#) .

**Head file reference**

fpd\_pageobjTempl.h: 972

**FPDGraphStateNew****Syntax**

```
FPD_GraphState FPDGraphStateNew (void );
```

**Description**

Creates a new empty PDF graphic state object.

**Return**

A new empty PDF graphic state object.

**Head file reference**

fpd\_pageobjTempl.h: 831

**FPDGraphStateSetDashArray****Syntax**

```
void FPDGraphStateSetDashArray (
    FPD\_GraphState graphState,
    FS\_FLOAT* dashArray
);
```

**Description**

Sets the dash phase for line dash pattern.

**Parameter**

---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

---

dashArray	[In] The input dash array.
-----------	----------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 991

**Since**[SDK LATEST VERSION > 1.0](#)**FPDGraphStateSetDashCount****Syntax**

```
void FPDGraphStateSetDashCount (
    FPD\_GraphState graphState,
    FS\_INT32 count
);
```

**Description**

In order to keep heap integrity, the function is used to allocate enough buffer for dash array.

**Parameter**

---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

---

count	[In] The new count of dash points.
-------	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 849

**FPDGraphStateSetDashPhase****Syntax**

```
void FPDGraphStateSetDashPhase (
    FPD\_GraphState graphState,
    FS\_FLOAT dashPhase
);
```

**Description**

Sets the new dash phase for line dash pattern.

**Parameter**

---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

dashPhase	[In] The new dash phase.
-----------	--------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 886

**FPDGraphStateSetLineCap****Syntax**

```
void FPDGraphStateSetLineCap (
    FPD\_GraphState graphState,
    FPD\_LineCap cap
);
```

**Description**

Sets the new line cap style.

**Parameter**


---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

cap	[In] The new style of line cap.
-----	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 905

**FPDGraphStateSetLineJoin****Syntax**

```
void FPDGraphStateSetLineJoin (
    FPD\_GraphState graphState,
    FPD\_LineJoin join
);
```

**Description**

Sets the new style of line join.

**Parameter**

---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

join	[In] The new style of line join.
------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 924

**FPDGraphStateSetLineWidth****Syntax**

```
void FPDGraphStateSetLineWidth (
    FPD\_GraphState graphState,
    FS\_FLOAT width
);
```

**Description**

Sets new width of lines.

**Parameter**


---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

width	[In] The new width of lines.
-------	------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 962

**FPDGraphStateSetMiterLimit****Syntax**

```
void FPDGraphStateSetMiterLimit (
    FPD\_GraphState graphState,
    FS\_FLOAT limit
);
```

**Description**

Sets the new miter limit for line join.

**Parameter**

---

graphState	[In] The input graphic state object.
------------	--------------------------------------

---

limit	[In] The new miter limit for line join.
-------	---

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 943

## FPD\_IconFit

**[Return from Used by](#)**

### Description

A FPD\_IconFit represents a set of icon that specifying how to display the widget otation's icon within its annotation rectangle. See [FPDIconFitNew](#) , [FPDIconFitDestroy](#) .

### Returned from

**[FPDIconFitNew](#)**

### Used by

[FPDFFormControlGetIconFit](#)  
[FPDIconFitDestroy](#)  
[FPDIconFitGetDict](#)  
[FPDIconFitGetFittingBounds](#)  
[FPDIconFitGetIconPosition](#)  
[FPDIconFitGetScaleMethod](#)  
[FPDIconFitIsProportionalScale](#)  
[FPDIconFitProportionalScale](#)  
[FPDIconFitSetFittingBounds](#)  
[FPDIconFitSetIconPosition](#)  
[FPDIconFitSetScaleMethod](#)

## Functions

### Functions summary

**[FPDIconFitDestroy](#)**

Destroys a PDF icon fit.

**[FPDIconFitGetDict](#)**

Gets the icon fit dictionary.

**[FPDIconFitGetFittingBounds](#)**

Gets the fitting-bound flag which indicates that the button appearance should be scaled to fit fully within the bounds of the annotation without taking into consideration the line width of the border.

**FPDIconFitGetIconPosition**

Gets the the fraction of leftover space to allocate at the left and bottom of the icon.

**FPDIconFitGetScaleMethod**

Gets the scale method.

**FPDIconFitIsProportionalScale**

Checks whether the scaling is proportional.

**FPDIconFitNew**

Creates a PDF icon fit from a PDF dictionary.

**FPDIconFitProportionalScale**

Change the proportional scaling flag.

**FPDIconFitSetFittingBounds**

Sets the fitting-bound flag.

**FPDIconFitSetIconPosition**

Sets the the fraction of leftover space to allocate at the left and bottom of the icon.

**FPDIconFitSetScaleMethod**

Sets the scale method.

## Functions detail

**FPDIconFitDestroy****Syntax**

```
void FPDIconFitDestroy (
    FPD\_IconFit iconFit
);
```

**Description**

Destroys a PDF icon fit.

**Parameter**

---

iconFit	[In] The input PDF icon fit.
---------	------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5406

**FPDIconFitGetDict****Syntax**

```
FPD_Object FPDIconFitGetDict (
    FPD\_IconFit iconFit
);
```

**Description**

Gets the icon fit dictionary.

**Parameter**

---

iconFit	[In] The input PDF icon fit.
---------	------------------------------

---

**Return**

The icon fit dictionary.

**Head file reference**

fpd\_docTempl.h: 5496

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FPDIconFitGetFittingBounds****Syntax**

```
FS_BOOL FPDIconFitGetFittingBounds (
    FPD\_IconFit iconFit
);
```

**Description**

Gets the fitting-bound flag which indicates that the button appearance should be scaled to fit fully within the bounds of the annotation without taking into consideration the line width of the border.

**Parameter**

---

iconFit	[In] The input PDF icon fit.
---------	------------------------------

---

**Return**

The fitting-bound flag

**Head file reference**

fpd\_docTempl.h: 5475

**FPDIconFitGetIconPosition****Syntax**

```
void FPDIconFitGetIconPosition (
    FPD\_IconFit iconFit,
    FS\_FLOAT* outLeft,
    FS\_FLOAT* outBottom
);
```

**Description**

Gets the the fraction of leftover space to allocate at the left and bottom of the icon.

**Parameter**

iconFit	[In] The input PDF icon fit.
outLeft	[Out] It receives the x-direction fraction.
outBottom	[Out] It receives the y-direction fraction.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5453

**FPDIconFitGetScaleMethod****Syntax**

```
FS_INT32 FPDIconFitGetScaleMethod (
    FPD\_IconFit iconFit
);
```

**Description**

Gets the scale method.

**Parameter**

iconFit	[In] The input PDF icon fit.
---------	------------------------------

**Return**

The scale method.

**Head file reference**

fpd\_docTempl.h: 5415

**FPDIconFitIsProportionalScale****Syntax**

```
FS_BOOL FPDIconFitIsProportionalScale (
    FPD\_IconFit iconFit
);
```

**Description**

Checks whether the scaling is proportional.

**Parameter**

---

iconFit	[In] The input PDF icon fit.
---------	------------------------------

---

**Return**

[TRUE](#) for the scaling being proportional.

**Head file reference**

fpd\_docTempl.h: 5434

**FPDIconFitNew****Syntax**

```
FPD_IconFit FPDIconFitNew (
    FPD\_Object dict
);
```

**Description**

Creates a PDF icon fit from a PDF dictionary.

**Parameter**

---

dict	[In] The input PDF dictionary.
------	--------------------------------

---

**Return**

A PDF icon fit.

**Head file reference**

fpd\_docTempl.h: 5397

**FPDIconFitProportionalScale****Syntax**

```
void FPDIconFitProportionalScale (
    FPD\_IconFit iconFit,
    FS\_BOOL bScale
);
```

**Description**

Change the proportional scaling flag.

**Parameter**

---

iconFit	[In] The input PDF icon fit.
---------	------------------------------

---

---

bScale	[In] Whether the proportional scaling flag to be set.
--------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5443

**FPDIconFitSetFittingBounds****Syntax**

```
void FPDIconFitSetFittingBounds (
    FPD\_IconFit iconFit,
    FS\_BOOL bFitBounds
);
```

**Description**

Sets the fitting-bound flag.

**Parameter**

---

iconFit	[In] The input PDF icon fit.
---------	------------------------------

---

---

bFitBounds	[In] The input fitting bound flag.
------------	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5486

**FPDIconFitSetIconPosition****Syntax**

```
void FPDIconFitSetIconPosition (
    FPD\_IconFit iconFit,
    FS\_FLOAT fLeft,
    FS\_FLOAT fBottom
);
```

**Description**

Sets the the fraction of leftover space to allocate at the left and bottom of the icon.

**Parameter**

---

iconFit	[In] The input PDF icon fit.
---------	------------------------------

---

---

fLeft	[In] The input x-direction fraction.
-------	--------------------------------------

---

fBottom	[In] The input y-direction fraction.
---------	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5464

**FPDIconFitSetScaleMethod****Syntax**

```
void FPDIconFitSetScaleMethod (
    FPD\_IconFit iconFit,
    FPD\_IconScaleMethod eScaleFunction
);
```

**Description**

Sets the scale method.

**Parameter**


---

iconFit	[In] The input PDF icon fit.
---------	------------------------------

---

eScaleFunction	[In] The new scale method.
----------------	----------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 5424

**FPD\_Image****[Return from Used by](#)****Description**

Holding image(sampled image) data . see *CHAPTER 4.8 in PDF reference* . A sampled image (or just image for short) is a rectangular array of sample values, each representing a color. It can be an *XObject* image, or an *inline* image. See [FPDImageNew](#) , [FPDImageDestroy](#) .

**Returned from**

---

[FPDDocLoadImageF](#)  
[FPDImageObjectGetImage](#)  
[FPDImageClone](#)  
[FPDImageNew](#)

### Used by

[FPDImageObjectSetImage](#)  
[FPDImageClone](#)  
[FPDImageDestroy](#)  
[FPDImageGetDIBSource](#)  
[FPDImageGetDict](#)  
[FPDImageGetDocument](#)  
[FPDImageGetMask](#)  
[FPDImageGetOC](#)  
[FPDImageGetPixelHeight](#)  
[FPDImageGetPixelWidth](#)  
[FPDImageGetStream](#)  
[FPDImageIsInterpol](#)  
[FPDImageIsMask](#)  
[FPDImageLoadDIBitmap](#)  
[FPDImageLoadDIBitmapProgressive](#)  
[FPDImageLoadImageF](#)  
[FPDImageResetCache](#)  
[FPDImageSetImage](#)  
[FPDImageSetJpegImage](#)

## Structures

### Structures summary

#### [FPD\\_MeshVertex](#)

No document exits.

### Structs detail

#### FPD\_MeshVertex

##### Syntax

```
typedef struct __FPD_MeshVertex__{
    FS\_FLOAT x, y,
    FS\_FLOAT r, g, b
}FPD_MeshVertex, *PFPD_MeshVertex;
```

##### Description

No document exits.

##### Head file reference

fpd\_resourceExpT.h: 348

##### x, y

No document exits.

##### r, g, b

No document exists.

## Functions

### Functions summary

#### [\*\*FPDImageClone\*\*](#)

Clones an image.

#### [\*\*FPDImageDestroy\*\*](#)

Destroys the PDF image data object.

#### [\*\*FPDImageGetDIBSource\*\*](#)

Gets the DIB source.

#### [\*\*FPDImageGetDict\*\*](#)

Gets the image dictionary.

#### [\*\*FPDImageGetDocument\*\*](#)

Gets the document.

#### [\*\*FPDImageGetMask\*\*](#)

Gets the mask.

#### [\*\*FPDImageGetMatteColor\*\*](#)

Gets the matte color. Since: [SDK\\_LATEEST\\_VERSION](#) > 2.1.0.3

\*\*\*\*\*

#### [\*\*FPDImageGetOC\*\*](#)

Gets the optional content dictionary.

#### [\*\*FPDImageGetPixelHeight\*\*](#)

Gets the pixel height.

#### [\*\*FPDImageGetPixelWidth\*\*](#)

Gets the pixel width.

#### [\*\*FPDImageGetStream\*\*](#)

Gets the image stream.

#### [\*\*FPDImageIsInterpol\*\*](#)

Checks whether image interpolation is to be performed.

#### [\*\*FPDImageIsMask\*\*](#)

Checks whether the image is a mask.

#### [\*\*FPDImageLoadDIBitmap\*\*](#)

Loads DIB source of the image. Optionally the mask info can be returned as well. You need invoke FSDIBitmapClone to generate the buffer for bitmap.

#### [\*\*FPDImageLoadDIBitmapProgressive\*\*](#)

Loads DIB source of the image progressively. You need invoke FSDIBitmapClone to generate the buffer for bitmap.

#### [\*\*FPDImageLoadImageF\*\*](#)

Loads from an image stream.

#### [\*\*FPDImageNew\*\*](#)

Creates a new PDF image data object.

#### [\*\*FPDImageResetCache\*\*](#)

Resets the page cache.

#### [\*\*FPDImageSetImage\*\*](#)

Changes image data from a DIB.

#### [\*\*FPDImageSetJpegImage\*\*](#)

Changes image data from a JPEG encoded block.

## Functions detail

### FPDImageClone

#### Syntax

```
FPD_Image FPDImageClone (
    FPD Image image
);
```

#### Description

Clones an image.

#### Parameter

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

#### Return

The clone of the image.

#### Head file reference

fpd\_resourceTempl.h: 1997

### FPDImageDestroy

#### Syntax

```
void FPDImageDestroy (
    FPD Image image
);
```

#### Description

Destroys the PDF image data object.

#### Parameter

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

#### Return

void

#### Head file reference

fpd\_resourceTempl.h: 1978

### FPDImageGetDIBSource

#### Syntax

```
FS_DIBitmap FPDImageGetDIBSource (
    FPD Image image
);
```

);

**Description**

Gets the DIB source.

**Parameter**

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

**Return**

The DIB source.

**Head file reference**

fpd\_resourceTempl.h: 2143

**Since**

[SDK LATEEST VERSION > 2.1.0.3](#)

**FPDImageGetDict****Syntax**

```
FPD_Object FPDImageGetDict (
    FPD\_Image image
);
```

**Description**

Gets the image dictionary.

**Parameter**

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

**Return**

The image dictionary.

**Head file reference**

fpd\_resourceTempl.h: 2015

**FPDImageGetDocument****Syntax**

```
FPD_Document FPDImageGetDocument (
    FPD\_Image image
);
```

**Description**

Gets the document.



**Parameter**

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

**Return**

The document.

**Head file reference**

fpd\_resourceTempl.h: 2033

**FPDImageGetMask****Syntax**

```
FS_DIBitmap FPDImageGetMask (
    FPD Image image
);
```

**Description**

Gets the mask.

**Parameter**

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

**Return**

The mask.

**Head file reference**

fpd\_resourceTempl.h: 2153

**Since**

[SDK LATEST VERSION > 2.1.0.3](#)

**FPDImageGetMatteColor****Syntax**

```
FPDImageGetMatteColor (
    image
);
```

**Description**

Gets the matte color. Since: [SDK LATEST VERSION > 2.1.0.3](#)

\*\*\*\*\*

**Parameter**

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

**Return**

The matte color.

**Head file reference**

fpd\_resourceTempl.h: 2163

**FPDImageGetOC****Syntax**

```
FPD_Object FPDImageGetOC (
    FPD\_Image image
);
```

**Description**

Gets the optional content dictionary.

**Parameter**

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

**Return**

The optional content dictionary.

**Head file reference**

fpd\_resourceTempl.h: 2024

**FPDImageGetPixelHeight****Syntax**

```
FS_INT32 FPDImageGetPixelHeight (
    FPD\_Image image
);
```

**Description**

Gets the pixel height.

**Parameter**

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

**Return**

The pixel height.

**Head file reference**

fpd\_resourceTempl.h: 2042

## FPDImageGetPixelWidth

### Syntax

```
FS_INT32 FPDImageGetPixelWidth (
    FPD\_Image image
);
```

### Description

Gets the pixel width.

### Parameter

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

### Return

The pixel width.

### Head file reference

fpd\_resourceTempl.h: 2051

## FPDImageGetStream

### Syntax

```
FPD_Object FPDImageGetStream (
    FPD\_Image image
);
```

### Description

Gets the image stream.

### Parameter

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

### Return

The image stream.

### Head file reference

fpd\_resourceTempl.h: 2006

## FPDImageIsInterpol

### Syntax

```
FS_BOOL FPDImageIsInterpol (
```

```
FPD_Image image  
);
```

**Description**

Checks whether image interpolation is to be performed.

**Parameter**

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

**Return**

[TRUE](#) if the image interpolation is to be performed, otherwise [FALSE](#).

**Head file reference**

fpd\_resourceTempl.h: 2069

**FPDImageIsMask****Syntax**

```
FS_BOOL FPDImageIsMask (  
    FPD_Image image  
) ;
```

**Description**

Checks whether the image is a mask.

**Parameter**

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

**Return**

[TRUE](#) if the image is a mask, otherwise [FALSE](#).

**Head file reference**

fpd\_resourceTempl.h: 2060

**FPDImageLoadDIBitmap****Syntax**

```
FS_DIBitmap FPDImageLoadDIBitmap (  
    FPD_Image image,  
    FS_DIBitmap* outMask,  
    FS_DWORD* outMatteColor,  
    FS_BOOL bStdCS,  
    FS_DWORD GroupFamily,  
    FS_BOOL bLoadMask  
) ;
```

**Description**

Loads DIB source of the image. Optionally the mask info can be returned as well. You need invoke FSDIBitmapClone to generate the buffer for bitmap.

**Parameter**

image	[In] The input PDF image data object.
outMask	[Out] It receives the mask of the image.
outMatteColor	[Out] It receives the matte color.
bStdCS	[In] The default value is <a href="#">FALSE</a> . Indicates whether use standard colorspace conversion(CMYK->RGB) or not.
GroupFamily	[In] The default value is 0. The group color space family for whether to adopt a special algorithm, and the group is a group or form include this image source.
bLoadMask	[In] The default value is <a href="#">FALSE</a> . when processing luminosity to mask alpha, adopt a special algorithm.

**Return**

The DIB source constructed.

**Head file reference**

fpd\_resourceTempl.h: 2078

**Note:** The result bitmaps are NOT cached, so the caller must release them when finished using.

**FPDImageLoadDIBitmapProgressive****Syntax**

```
FS_BOOL FPDImageLoadDIBitmapProgressive (
    FPD\_Image image,
    FPD\_Object formResourceDict,
    FPD\_Object pageResourceDict,
    FS_BOOL bStdCS,
    FS_DWORD GroupFamily,
    FS_BOOL bLoadMask
);
```

**Description**

Loads DIB source of the image progressively. You need invoke FSDIBitmapClone to generate the buffer for bitmap.

**Parameter**


---

image	[In] The input PDF image data object.
formResourceDict	[In] The input form resource dictionary.
pageResourceDict	[In] The input page resource dictionary.
bStdCS	[In] The default value is <a href="#">FALSE</a> . Indicates whether use standard colorspace conversion(CMYK->RGB) or not.
GroupFamily	[In] The default value is 0. The group color space family for whether to adopt a special algorithm, and the group is a group or form include this image source.
bLoadMask	[In] The default value is <a href="#">FALSE</a> . when processing luminosity to mask alpha, adopt a special algorithm.

---

**Return**

TRUE for success, otherwise failure.

**Head file reference**

fpd\_resourceTempl.h: 2128

**Note:** The result bitmaps are NOT cached, so the caller must release them when finished using.

**Since**

[SDK LATEEST VERSION > 2.1.0.3](#)

**FPDImageLoadImageF****Syntax**

```
FS_BOOL FPDImageLoadImageF (
    FPD\_Image image,
    FPD\_Object imageStream
);
```

**Description**

Loads from an image stream.

**Parameter**

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

imageStream	[In] The input image stream.
-------------	------------------------------

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_resourceTempl.h: 1987

**FPDImageNew****Syntax**

```
FPD_Image FPDImageNew (
    FPD\_Document doc
);
```

**Description**

Creates a new PDF image data object.

**Parameter**

---

doc	[In] The PDF document.
-----	------------------------

---

**Return**

A new PDF image data object.

**Head file reference**

fpd\_resourceTempl.h: 1969

**FPDImageResetCache****Syntax**

```
void FPDImageResetCache (
    FPD\_Image image,
    FPD\_Page page,
    const FS\_DIBitmap DIBitmap
);
```

**Description**

Resets the page cache.

**Parameter**

---

image	[In] The input PDF image data object.
-------	---------------------------------------

---

page	[In] The page.
------	----------------

DIBitmap	[In] The new DIB to set.
----------	--------------------------

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 2100

**FPDImageSetImage****Syntax**

```
void FPDImageSetImage (
    FPD_Image image,
    const FS_DIBitmap DIBitmap,
    FS_BOOL bCompress,
    FS_DIBitmap pMask
);
```

**Description**

Changes image data from a DIB.

**Parameter**

image	[In] The input PDF image data object.
-------	---------------------------------------

DIBitmap	[In] The DIB data.
----------	--------------------

bCompress	[In] Whether to compress the DIB.
-----------	-----------------------------------

pMask	[In] Mask image, it's valid only <i>DIBitmap</i> has on alpha channel. Mask image should be <i>FS_DIB_1bppMask</i> or <i>FS_DIB_8bppMask</i> .
-------	--

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 2092

**Note:** The bitmaps stored in the page cache must be reset by *FPDImageResetCache()* for all loaded pages.

**FPDImageSetJpegImage**

**Syntax**

```
void FPDImageSetJpegImage (
    FPD_Image image,
    FS_BYTE* pImageDataBuf,
    FS_DWORD size
);
```

**Description**

Changes image data from a JPEG encoded block.

**Parameter**

image	[In] The input PDF image data object.
-------	---------------------------------------

pImageDataBuf	[In] The JPEG data.
---------------	---------------------

size	[In] The size in bytes of the JPEG data.
------	--

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 2105

**Note:** The bitmaps stored in the page cache must be reset by FPDImageResetCache() for all loaded pages.

## FPD\_ImageObject

### Description

A FPD\_ImageObject is a sampled image or image mask, and corresponds to a PDF Image resource (see *Stencil Masking in Section 4.8, Images, in the PDF Reference* ).

### Functions

#### Functions summary

##### [FPDImageObjectCalcBoundingBox](#)

Calculates the bounding box.

##### [FPDImageObjectDestroy](#)

Destroys the PDF image object.

##### [FPDImageObjectGetImage](#)

Gets the image data object.

##### [FPDImageObjectGetTransformMatrix](#)

Gets the transformation matrix.

**[FPDImageObjectNew](#)**

Creates a new empty PDF image object.

**[FPDImageObjectSetImage](#)**

Sets the image data object.

**[FPDImageObjectSetTransformMatrix](#)**

Sets the transformation matrix.

**[FPDImageObjectTransform](#)**

Transforms the image object.

## Functions detail

**FPDImageObjectCalcBoundingBox****Syntax**

```
void FPDImageObjectCalcBoundingBox (
    FPD PageObject objImage
);
```

**Description**

Calculates the bounding box.

**Parameter**

---

objImage	[In] The input PDF image object.
----------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1641

**FPDImageObjectDestroy****Syntax**

```
void FPDImageObjectDestroy (
    FPD PageObject objImage
);
```

**Description**

Destroys the PDF image object.

**Parameter**

---

objImage	[In] The input PDF image object.
----------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1622

**FPDImageObjectGetImage****Syntax**

```
FPD_Image FPDImageObjectGetImage (
    FPD\_PageObject objImage
);
```

**Description**

Gets the image data object.

**Parameter**

---

objImage	[In] The input PDF image object.
----------	----------------------------------

---

**Return**

The image data object.

**Head file reference**

fpd\_pageobjTempl.h: 1670

**FPDImageObjectGetTransformMatrix****Syntax**

```
void FPDImageObjectGetTransformMatrix (
    FPD\_PageObject objImage,
    FS\_AffineMatrix* outmatrix
);
```

**Description**

Gets the transformation matrix.

**Parameter**

---

objImage	[In] The input PDF image object.
----------	----------------------------------

---

outmatrix	[Out] It receives the transformation matrix.
-----------	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1650

## FPDImageObjectNew

### Syntax

```
FPD_PageObject FPDImageObjectNew (void );
```

### Description

Creates a new empty PDF image object.

### Return

A new empty PDF image object.

### Head file reference

fpd\_pageobjTempl.h: 1613

## FPDImageObjectSetImage

### Syntax

```
void FPDImageObjectSetImage (
    FPD\_PageObject objImage,
    FPD\_Image image
);
```

### Description

Sets the image data object.

### Parameter

---

objImage	[In] The input PDF image object.
----------	----------------------------------

---

image	[In] The input image data object.
-------	-----------------------------------

---

### Return

void

### Head file reference

fpd\_pageobjTempl.h: 1679

## FPDImageObjectSetTransformMatrix

### Syntax

```
void FPDImageObjectSetTransformMatrix (
    FPD\_PageObject objImage,
    const FS\_AffineMatrix\* matrix
);
```

### Description

Sets the transformation matrix.

**Parameter**

---

objImage	[In] The input PDF image object.
----------	----------------------------------

---

matrix	[In] The transformation matrix.
--------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1660

**FPDImageObjectTransform****Syntax**

```
void FPDImageObjectTransform (
    FPD\_PageObject objImage,
    FS\_AffineMatrix matrix
);
```

**Description**

Transforms the image object.

**Parameter**

---

objImage	[In] The input PDF image object.
----------	----------------------------------

---

matrix	[In] The transformation matrix.
--------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1631

## FPD\_INLINEIMAGES

### Description

A FPD\_INLINEIMAGES is an image whose data is stored in the page description's contents stream instead of being stored as an image resource (see FPD\_ImageObject).

FPD\_INLINEIMAGES is a subclass of FPD\_PageObject and corresponds to the PDF inline image operator ( see *Section 4.8.6, In-Line Images, in the PDF Reference* ).

Inline images generally are used for images with small amounts of data (up to several kilobytes), while image resources are used for large images.

## Functions

### Functions summary

#### [FPDInlineImagesAddMatrix](#)

Adds a transform matrix.

#### [FPDInlineImagesCountMatrix](#)

Counts the number of transform matrix in this object.

#### [FPDInlineImagesDestroy](#)

Destroys the PDF inline images object.

#### [FPDInlineImagesGetMatrix](#)

Gets a transform matrix from object by index.

#### [FPDInlineImagesGetStream](#)

Gets the stream.

#### [FPDInlineImagesNew](#)

Creates a new empty PDF inline images object.

#### [FPDInlineImagesSetStream](#)

Sets the stream

### Functions detail

#### [FPDInlineImagesAddMatrix](#)

##### **Syntax**

```
void FPDInlineImagesAddMatrix (
    FPD_PageObject objInlineImgs,
    const FS_AffineMatrix* matrix
);
```

##### **Description**

Adds a transform matrix.

##### **Parameter**

---

objInlineImgs	[In] The input PDF inline images object.
---------------	--

---

matrix	[In] The input transform matrix.
--------	----------------------------------

---

##### **Return**

void

##### **Head file reference**

fpd\_pageobjTempl.h: 1902

## FPDInlineImagesCountMatrix

### Syntax

```
int FPDInlineImagesCountMatrix (
    FPD\_PageObject objInlineImgs
);
```

### Description

Counts the number of transform matrix in this object.

### Parameter

---

objInlineImgs	[In] The input PDF inline images object.
---------------	--

---

### Return

The number of transform matrix in this object.

### Head file reference

fpd\_pageobjTempl.h: 1912

## FPDInlineImagesDestroy

### Syntax

```
void FPDInlineImagesDestroy (
    FPD\_PageObject objInlineImgs
);
```

### Description

Destroys the PDF inline images object.

### Parameter

---

objInlineImgs	[In] The input PDF inline images object.
---------------	--

---

### Return

void

### Head file reference

fpd\_pageobjTempl.h: 1893

## FPDInlineImagesGetMatrix

### Syntax

```
FS_AffineMatrix FPDInlineImagesGetMatrix (
    FPD\_PageObject objInlineImgs,
    FS\_INT32 index
);
```

**Description**

Gets a transform matrix from object by index.

**Parameter**

---

objInlineImgs	[In] The input PDF inline images object.
---------------	--

---

---

index	[In] The index of transform matrix.
-------	-------------------------------------

---

**Return**

The transform matrix from object by index.

**Head file reference**

fpd\_pageobjTempl.h: 1921

**FPDInlineImagesGetStream****Syntax**

```
FPD_Object FPDInlineImagesGetStream (
    FPD PageObject objInlineImgs
);
```

**Description**

Gets the stream.

**Parameter**

---

objInlineImgs	[In] The input PDF inline images object.
---------------	--

---

**Return**

The stream.

**Head file reference**

fpd\_pageobjTempl.h: 1931

**FPDInlineImagesNew****Syntax**

```
FPD_PageObject FPDInlineImagesNew (void );
```

**Description**

Creates a new empty PDF inline images object.

**Return**

A new empty PDF inline images object.

**Head file reference**

fpd\_pageobjTempl.h: 1884

**FPDInlineImagesSetStream****Syntax**

```
void FPDInlineImagesSetStream (
    FPD PageObject objInlineImgs,
    FPD Object stream
);
```

**Description**

Sets the stream

**Parameter**

---

objInlineImgs	[In] The input PDF inline images object.
---------------	--

---

stream	[In] The input stream
--------	-----------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1940

## FPD\_InterForm

**[Return from Used by](#)****Description**

A PDF interactive form (AcroForm) object. It is a collection of fields for gathering formation interactively form user. A FPD\_Document object may contain any number of fields ppearing on any combination of pages, all of which make up a single, global interactive orm spanning the entire document.

**Returned from**

[FPDFFormControlGetInterForm](#)  
[FPDFFormFieldGetInterForm](#)  
[FPDInterFormNew](#)

**Used by**

[FPDInterFormAddControl](#)

[FPDInterFormAddFormFont](#)  
[FPDInterFormAddNativeFormFont](#)  
[FPDInterFormAddNativeFormFont2](#)  
[FPDInterFormCheckRequiredFields](#)  
[FPDInterFormClearUpdatedFlag](#)  
[FPDInterFormCountControls](#)  
[FPDInterFormCountFields](#)  
[FPDInterFormCountFieldsInCalculationOrder](#)  
[FPDInterFormCountFormFonts](#)  
[FPDInterFormCountInternalFields](#)  
[FPDInterFormCountPageControls](#)  
[FPDInterFormDeleteControl](#)  
[FPDInterFormDeleteField](#)  
[FPDInterFormDeleteField2](#)  
[FPDInterFormDestroy](#)  
[FPDInterFormExportToFDF](#)  
[FPDInterFormExportToFDF2](#)  
[FPDInterFormFindFieldInCalculationOrder](#)  
[FPDInterFormFindFormFont](#)  
[FPDInterFormFindFormFont2](#)  
[FPDInterFormFindFormFont3](#)  
[FPDInterFormFixPageFields](#)  
[FPDInterFormGetAllFieldNames](#)  
[FPDInterFormGetControl](#)  
[FPDInterFormGetControlAtPoint](#)  
[FPDInterFormGetControlByDict](#)  
[FPDInterFormGetDefaultAppearance](#)  
[FPDInterFormGetDefaultFormFont](#)  
[FPDInterFormGetDocument](#)  
[FPDInterFormGetField](#)  
[FPDInterFormGetFieldByDict](#)  
[FPDInterFormGetFieldInCalculationOrder](#)  
[FPDInterFormGetFormAlignment](#)  
[FPDInterFormGetFormDict](#)  
[FPDInterFormGetFormFont](#)  
[FPDInterFormGetFormFont2](#)  
[FPDInterFormGetFormFont3](#)  
[FPDInterFormGetFormNotify](#)  
[FPDInterFormGetInternalField](#)  
[FPDInterFormGetNativeFormFont](#)  
[FPDInterFormGetNativeFormFont2](#)  
[FPDInterFormGetPageControl](#)  
[FPDInterFormGetPageWithWidget](#)  
[FPDInterFormImportFromFDF](#)  
[FPDInterFormInsertFieldInCalculationOrder](#)  
[FPDInterFormIsUpdated](#)  
[FPDInterFormIsValidFormControl](#)  
[FPDInterFormIsValidFormField](#)  
[FPDInterFormMoveFieldInCalculationOrder](#)  
[FPDInterFormNeedConstructAP](#)  
[FPDInterFormNeedConstructAP2](#)  
[FPDInterFormNewControl](#)  
[FPDInterFormNewField](#)  
[FPDInterFormReloadForm](#)  
[FPDInterFormRemoveFieldInCalculationOrder](#)  
[FPDInterFormRemoveFormFont](#)  
[FPDInterFormRemoveFormFont2](#)

[FPDInterFormRenameControl](#)  
[FPDInterFormRenameField](#)  
[FPDInterFormRenameField2](#)  
[FPDInterFormResetForm](#)  
[FPDInterFormResetForm2](#)  
[FPDInterFormSetDefaultAppearance](#)  
[FPDInterFormSetDefaultFormFont](#)  
[FPDInterFormSetFormAlignment](#)  
[FPDInterFormSetFormNotify](#)  
[FPDInterFormValidateFieldName](#)  
[FPDInterFormValidateFieldName2](#)  
[FPDInterFormValidateFieldName3](#)

## Definitions

### Definitions summary

#### [FPD\\_CLRTYPE\\_CMYK](#)

CMYK.

#### [FPD\\_CLRTYPE\\_GRAY](#)

Gray.

#### [FPD\\_CLRTYPE\\_RGB](#)

RGB.

#### [FPD\\_CLRTYPE\\_TRANSPARENT](#)

Transparent.

### Definitions detail

#### FPD\_CLRTYPE\_CMYK

##### Syntax

```
#define FPD_CLRTYPE_CMYK 3
```

##### Description

CMYK.

##### Group

[FPDInterFormColorTypes](#)

##### Head file reference

fpd\_docExpT.h: 1038

#### FPD\_CLRTYPE\_GRAY

##### Syntax

```
#define FPD_CLRTYPE_GRAY 1
```

##### Description

Gray.

##### Group

[FPDInterFormColorTypes](#)**Head file reference**

fpd\_docExpT.h: 1034

**FPD\_CLRTYPE\_RGB****Syntax**

#define FPD\_CLRTYPE\_RGB 2

**Description**

RGB.

**Group**[FPDInterFormColorTypes](#)**Head file reference**

fpd\_docExpT.h: 1036

**FPD\_CLRTYPE\_TRANSPARENT****Syntax**

#define FPD\_CLRTYPE\_TRANSPARENT 0

**Description**

Transparent.

**Group**[FPDInterFormColorTypes](#)**Head file reference**

fpd\_docExpT.h: 1032

## Functions

### Functions summary

[FPDInterFormAddControl](#)

Adds a control to the PDF interactive form with a widget.

[FPDInterFormAddFormFont](#)

Adds a form font with specified name tag.

[FPDInterFormAddNativeFont](#)

Adds a native font with a character set.

[FPDInterFormAddNativeFont2](#)

Adds the native font of the PDF document.

[FPDInterFormAddNativeFormFont](#)

Adds a native form font with specified name tag.

**FPDInterFormAddNativeFormFont2**

Adds the native form font.

**FPDInterFormAddStandardFont**

Adds a standard font with font name.

**FPDInterFormAddSystemDefaultFont**

Adds the system default font.

**FPDInterFormAddSystemFont**

Adds a system font with font name and character set. ANSI version.

**FPDInterFormAddSystemFontW**

Adds a system font with font name and character set. Unicode version.

**FPDInterFormCheckRequiredFields**

Checks the required fields' checked flag.

**FPDInterFormClearUpdatedFlag**

Clears the updated flag.

**FPDInterFormCountControls**

Gets the count of specified controls.

**FPDInterFormCountFields**

Gets the count of specified fields.

**FPDInterFormCountFieldsInCalculationOrder**

The count of all fields in CO array.

**FPDInterFormCountFormFonts**

Gets the count of font resources.

**FPDInterFormCountInternalFields**

Gets the count of specified internal fields.

**FPDInterFormCountPageControls**

Gets the count of controls in specified page.

**FPDInterFormDeleteControl**

Deletes a control. control object will be deleted if success.

**FPDInterFormDeleteField**

Deletes a field with specified field name. If csName is empty, means for all fields.

**FPDInterFormDeleteField2**

Deletes a field. Field object will be deleted if success.

**FPDInterFormDestroy**

Destroys a PDF interactive form.

**FPDInterFormEnableUpdateAP**

Change the auto-update-appearance flag.

**FPDInterFormExportToFDF**

Exports the interactive form to a FDF format file.

**FPDInterFormExportToFDF2**

Exports the interactive form to a FDF format file, include or exclude some fields.

**FPDInterFormFindFieldInCalculationOrder**

Finds a form field in the CO array.

**FPDInterFormFindFormFont**

Finds a font with font pointer.

**FPDInterFormFindFormFont2**

Finds a font with font name. ANSI version.

**FPDInterFormFindFormFont3**

Finds a font with font name. Unicode version.

**FPDInterFormFixPageFields**

Supplement some form field from a page

**FPDInterFormGenerateNewResourceName**

Generate a new resource tag name, especially for ExtGState, ColorSpace, Font, etc.

**FPDInterFormGetAllFieldNames**

Gets all field names in the form.

**FPDInterFormGetControl**

Gets a form control.

**FPDInterFormGetControlAtPoint**

Retrieves the control by mouse position.

**FPDInterFormGetControlByDict**

Retrieves the control by widget dictionary.

**FPDInterFormGetDefaultAppearance**

Gets the default appearance interpreter.

**FPDInterFormGetDefaultFormFont**

Gets the default form font.

**FPDInterFormGetDocument**

Gets the PDF document.

**FPDInterFormGetField**

Gets a form field.

**FPDInterFormGetFieldByDict**

Retrieves the field by field dictionary.

**FPDInterFormGetFieldInCalculationOrder**

Gets a form field from the calculation array.

**FPDInterFormGetFormAlignment**

Gets the form alignment.

**FPDInterFormGetFormDict**

Gets the AcroForm dictionary.

**FPDInterFormGetFormFont**

Gets a PDF font from the font resources.

**FPDInterFormGetFormFont2**

Gets a PDF font with specified name tag.

**FPDInterFormGetFormFont3**

Finds a PDF font with specified font name.

**FPDInterFormGetFormNotify**

Gets the form notify handler.

**FPDInterFormGetInternalField**

Gets an internal field.

**FPDInterFormGetNativeFont**

Gets the native font name with a character set and a logical font structure.

**FPDInterFormGetNativeFont2**

Gets the native font name with a logical font structure.

**FPDInterFormGetNativeFormFont**

Gets a native-form font with a character set.

**FPDInterFormGetNativeFormFont2**

Gets the native-form font.

**FPDInterFormGetPageControl**

Gets a control in specified page.

**FPDInterFormGetPageWithWidget**

Retrieves the next or previous page for the page specified by iCurPage.

**FPDInterFormImportFromFDF**

Imports a FDF document.

**FPDInterFormInsertFieldInCalculationOrder**

Inserts a form field to the CO array.

**FPDInterFormIsUpdated**

Checks whether the form is updated.

**FPDInterFormIsValidFormControl**

Checks whether a pointer is a form control pointer.

**FPDInterFormIsValidFormField**

Checks whether a pointer is a form field pointer.

**FPDInterFormMoveFieldInCalculationOrder**

Moves a form field in the CO array to another position.

**FPDInterFormNeedConstructAP**

Checks whether to construct appearance streams and appearance dictionaries for all widget annotations in the document.

**FPDInterFormNeedConstructAP2**

Sets the construct flag.

**FPDInterFormNew**

Creates a form from a document. The bUpdateAP param specifies whether we need to regenerate appearance streams for the fields.

**FPDInterFormNewControl**

Creates and add a control in form, param iType uses FPD\_FORM\_FIELDTYPE\_X macros. inOutFieldName is in/out param. If succeed, inOutFieldName is the validated field full name.

**FPDInterFormNewField**

Creates and add a field in form, param iType uses FPD\_FORM\_FIELDTYPE\_X macros. inOutFieldName is in/out param. If succeed, csFieldName is the validated field full name.

**FPDInterFormReloadForm**

Reload the interactive form.

**FPDInterFormRemoveFieldInCalculationOrder**

Removes a field in the CO array.

**FPDInterFormRemoveFormFont**

Removes a form font with font pointer.

**FPDInterFormRemoveFormFont2**

Removes a form font with specified name tag.

**FPDInterFormRenameControl**

Rename a control's full name.

**FPDInterFormRenameField**

Rename a field's full name. The field identified by it's old full name.

**FPDInterFormRenameField2**

Rename a field's full name.

**FPDInterFormResetForm**

Resets form fields.

**FPDInterFormResetForm2**

Resets all form fields.

**FPDInterFormSetDefaultAppearance**

Sets the default appearance interpreter.

**FPDInterFormSetDefaultFormFont**

Sets the default form font.

**FPDInterFormSetFormAlignment**

Sets the form alignment.

**FPDInterFormSetFormNotify**

Sets the form notify handler.

**FPDInterFormUpdatingAPEnabled**

Checks whether need to update appearance automatically.

**FPDInterFormValidateFieldName**

Checks the field name of specified type is valid or not, receives the valid field name when it's invalid.

**FPDInterFormValidateFieldName2**

Checks the field name for the input field is valid or not, receives the valid field name when it's invalid.

**FPDInterFormValidateFieldName3**

Checks the field name for the input control is valid or not, receives the valid field name when it's invalid.

**Functions detail****FPDInterFormAddControl****Syntax**

```
FPD_FormControl FPDInterFormAddControl (
    FPD_InterForm form,
    const FPD_Object fieldDict,
    const FPD_Object pWidgetDict
);
```

**Description**

Adds a control to the PDF interactive form with a widget.

**Parameter**

form	[In] The input PDF interactive form.
fieldDict	[In] The input field dictionary.
pWidgetDict	[In] The input widget dictionary.

**Return**

A form control.

**Head file reference**

fpd\_docTempl.h: 4725

**Since**

[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

**FPDInterFormAddFormFont****Syntax**

```
void FPDInterFormAddFormFont (
    FPD_InterForm form,
    const FPD_Font font,
```

```
FS_ByteString* inOutNameTag  
);
```

**Description**

Adds a form font with specified name tag.

**Parameter**

form	[In] The input PDF interactive form.
font	[In] The input PDF font.
inOutNameTag	[In/Out] Input a name tag and receive a valid name tag of the font, such as /Helv (get rid of slash).

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4471

**FPDInterFormAddNativeFont****Syntax**

```
FPD_Font FPDInterFormAddNativeFont (  
    FS_BYTE iCharSet,  
    const FPD_Document doc  
) ;
```

**Description**

Adds a native font with a character set.

**Parameter**

iCharSet	[In] The input character set.
doc	[In] The PDF document.

**Return**

A PDF font.

**Head file reference**

fpd\_docTempl.h: 3974

## FPDInterFormAddNativeFont2

### Syntax

```
FPD_Font FPDInterFormAddNativeFont2 (
    const FPD_Document doc
);
```

### Description

Adds the native font of the PDF document.

### Parameter

---

doc	[In] The PDF document.
-----	------------------------

---

### Return

A PDF font.

### Head file reference

fpd\_docTempl.h: 3984

## FPDInterFormAddNativeFormFont

### Syntax

```
FPD_Font FPDInterFormAddNativeFormFont (
    FPD_InterForm form,
    FS_BYTE iCharSet,
    FS_ByteString* inOutNameTag
);
```

### Description

Adds a native form font with specified name tag.

### Parameter

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

---

iCharSet	[In] The input character set.
----------	-------------------------------

---

---

inOutNameTag	[In/Out] Input a name tag and receive a valid name tag of the font, such as /Helv (get rid of slash).
--------------	---

---

### Return

A PDF font.

### Head file reference

fpd\_docTempl.h: 4482

## FPDInterFormAddNativeFormFont2

### Syntax

```
FPD_Font FPDInterFormAddNativeFormFont2 (
    FPD\_InterForm form,
    FS\_ByteString* inOutNameTag
);
```

### Description

Adds the native form font.

### Parameter

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

inOutNameTag	[In/Out] Input a name tag and receive a valid name tag of the font, such as /Helv (get rid of slash).
--------------	---

---

### Return

A PDF font.

### Head file reference

fpd\_docTempl.h: 4493

## FPDInterFormAddStandardFont

### Syntax

```
FPD_Font FPDInterFormAddStandardFont (
    const FPD\_Document doc,
    FS\_LPSTR szFontName
);
```

### Description

Adds a standard font with font name.

### Parameter

---

doc	[In] The PDF document.
-----	------------------------

---

szFontName	[In] The input font name.
------------	---------------------------

---

### Return

A PDF font.

### Head file reference

fpd\_docTempl.h: 3943

## FPDInterFormAddSystemDefaultFont

### Syntax

```
FPD_Font FPDInterFormAddSystemDefaultFont (
    const FPD_Document doc
);
```

### Description

Adds the system default font.

### Parameter

---

doc	[In] The PDF document.
-----	------------------------

---

### Return

A PDF font.

### Head file reference

fpd\_docTempl.h: 3912

## FPDInterFormAddSystemFont

### Syntax

```
FPD_Font FPDInterFormAddSystemFont (
    const FPD_Document doc,
    FS_LPSTR szFontName,
    FS_BYTEx iCharSet
);
```

### Description

Adds a system font with font name and character set. ANSI version.

### Parameter

---

doc	[In] The PDF document.
-----	------------------------

---

szFontName	[In] The input font name.
------------	---------------------------

---

iCharSet	[In] The input character set.
----------	-------------------------------

---

### Return

A PDF font.

**Head file reference**

fpd\_docTempl.h: 3921

**FPDInterFormAddSystemFontW****Syntax**

```
FPD_Font FPDInterFormAddSystemFontW (
    const FPD_Document doc,
    FS_LPWSTR wszFontName,
    FS_BYTEx iCharSet
);
```

**Description**

Adds a system font with font name and character set. Unicode version.

**Parameter**

doc	[In] The PDF document.
wszFontName	[In] he input font name.
iCharSet	[In] The input character set.

**Return**

A PDF font.

**Head file reference**

fpd\_docTempl.h: 3932

**FPDInterFormCheckRequiredFields****Syntax**

```
FPD_FormField FPDInterFormCheckRequiredFields (
    FPD_InterForm form,
    const FS_PtrArray fields,
    FS_BOOL bIncludeOrExclude
);
```

**Description**

Checks the required fields' checked flag.

**Parameter**

form	[In] The input PDF interactive form.
------	--------------------------------------

fields	[In] The input form fields array. fields is the array of FPD_FormField.
--------	---

---

bIncludeOrExclude	[In] Whether to include or exclude the required fields.
-------------------	---

---

**Return**

The checked form field.

**Head file reference**

fpd\_docTempl.h: 4592

**FPDInterFormClearUpdatedFlag****Syntax**

```
void FPDInterFormClearUpdatedFlag (
    FPD\_InterForm form
);
```

**Description**

Clears the updated flag.

**Parameter**

form	[In] The input PDF interactive form.
------	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4706

**FPDInterFormCountControls****Syntax**

```
FS_DWORD FPDInterFormCountControls (
    FPD\_InterForm form,
    FS\_LPWSTR wszFieldName
);
```

**Description**

Gets the count of specified controls.

**Parameter**

form	[In] The input PDF interactive form.
------	--------------------------------------

---

---

wszFieldName	[In] The input field name to be matched. -empty means for all fields; or system will match the field name. -as shortest string, for example, text1 will match text1.0, text1.2.0, etc. but text1 will not match test10 or test11.1..
--------------	--

---

**Return**

The count of controls with specified field name to be matched.

**Head file reference**

fpd\_docTempl.h: 4106

**FPDInterFormCountFields****Syntax**

```
FS_DWORD FPDInterFormCountFields (
    FPD\_InterForm form,
    FS\_LPWSTR wszFieldName
);
```

**Description**

Gets the count of specified fields.

**Parameter**


---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

wszFieldName	[In] The input field name to be matched. -empty means for all fields; or system will match the field name. -as shortest string, for example, text1 will match text1.0, text1.2.0, etc. but text1 will not match test10 or test11.1..
--------------	--

---

**Return**

The count of fields with specified field name to be matched.

**Head file reference**

fpd\_docTempl.h: 4051

**FPDInterFormCountFieldsInCalculationOrder****Syntax**

```
FS_INT32 FPDInterFormCountFieldsInCalculationOrder (
    FPD\_InterForm form
);
```

**Description**

The count of all fields in CO array.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

**Return**

The count of all fields in CO array.

**Head file reference**

fpd\_docTempl.h: 4313

**FPDInterFormCountFormFonts****Syntax**

```
FS_DWORD FPDInterFormCountFormFonts (
    FPD_InterForm form
);
```

**Description**

Gets the count of font resources.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

**Return**

The count of font resources.

**Head file reference**

fpd\_docTempl.h: 4374

**FPDInterFormCountInternalFields****Syntax**

```
FS_DWORD FPDInterFormCountInternalFields (
    FPD_InterForm form,
    FS_LPWSTR wszFieldName
);
```

**Description**

Gets the count of specified internal fields.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

wszFieldName	[In] The input field name to be matched. -If empty, this will retrieve all internal field dictionaries from the root list of the whole form. -If has value, all internal fields which names match the csfieldName will return.
--------------	--

---

**Return**

The count of internal fields with specified field name to be matched.

**Head file reference**

fpd\_docTempl.h: 4251

**FPDInterFormCountPageControls****Syntax**

```
FS_INT32 FPDInterFormCountPageControls (
    FPD\_InterForm form,
    FPD\_Page page
);
```

**Description**

Gets the count of controls in specified page.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

The count of controls in specified page.

**Head file reference**

fpd\_docTempl.h: 4141

**FPDInterFormDeleteControl****Syntax**

```
void FPDInterFormDeleteControl (
    FPD\_InterForm form,
    FPD\_FormControl control
);
```

**Description**

Deletes a control. control object will be deleted if success.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

control	[In] The input control.
---------	-------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4239

**Note:** When delete controls, corresponding fields will be deleted only when they have no any control associated with them. If csFieldName is empty, it means to all fields or controls.

**FPDInterFormDeleteField****Syntax**

```
void FPDInterFormDeleteField (
    FPD\_InterForm form,
    FS\_LPWSTR wszFieldName
);
```

**Description**

Deletes a field with specified field name. If csName is empty, means for all fields.

**Parameter**


---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

wszFieldName	[In] The input field name.
--------------	----------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4217

**Note:** When delete fields, all controls assigned to them will be deleted too.

**FPDInterFormDeleteField2****Syntax**

```
void FPDInterFormDeleteField2 (
    FPD\_InterForm form,
    FPD\_FormField field
);
```

**Description**

Deletes a field. Field object will be deleted if success.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

field
-------

[In] The input field.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4228

**Note:** When delete fields, all controls assigned to them will be deleted too.

**FPDInterFormDestroy****Syntax**

```
void FPDInterFormDestroy (
    FPD_InterForm form
);
```

**Description**

Destroys a PDF interactive form.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3871

**FPDInterFormEnableUpdateAP****Syntax**

```
void FPDInterFormEnableUpdateAP (
    FS_BOOL bUpdateAP
);
```

**Description**

Change the auto-update-appearance flag.

**Parameter**

---

bUpdateAP	[In] the input auto-update-appearance flag.
-----------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3880

**FPDInterFormExportToFDF****Syntax**

```
FDF_Document FPDInterFormExportToFDF (
    FPD\_InterForm form,
    FS\_LPCWSTR szPDFPath
);
```

**Description**

Exports the interactive form to a FDF format file.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

szPDFPath	[In] The input FDF file full path.
-----------	------------------------------------

---

**Return**

A FDF document.

**Head file reference**

fpd\_docTempl.h: 4603

**FPDInterFormExportToFDF2****Syntax**

```
FDF_Document FPDInterFormExportToFDF2 (
    FPD\_InterForm form,
    FS\_LPCWSTR szPDFPath,
    FS\_PtrArray fields,
    FS\_BOOL bIncludeOrExclude
);
```

**Description**

---

Exports the interactive form to a FDF format file, include or exclude some fields.

#### Parameter

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

szPDFPath	[In] The input FDF file full path.
-----------	------------------------------------

---

fields	[In] The input form fields array. fields is the array of FPD_FormField.
--------	---

---

bIncludeOrExclude	[In] Whether to include or exclude the form fields.
-------------------	---

---

#### Return

A FDF document.

#### Head file reference

fpd\_docTempl.h: 4613

### FPDInterFormFindFieldInCalculationOrder

#### Syntax

```
FS_INT32 FPDInterFormFindFieldInCalculationOrder (
    FPD\_InterForm form,
    const FPD\_FormField field
);
```

#### Description

Finds a form field in the CO array.

#### Parameter

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

field	[In] The input form field.
-------	----------------------------

---

#### Return

The zero-based index of the specified field in the CO array. -1 means not found.

#### Head file reference

fpd\_docTempl.h: 4332

### FPDInterFormFindFormFont

**Syntax**

```
FS_BOOL FPDInterFormFindFormFont (
    FPD\_InterForm form,
    const FPD\_Font font,
    FS\_ByteString* outNameTag
);
```

**Description**

Finds a font with font pointer.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

font	[In] The input PDF font.
------	--------------------------

---

outNameTag	[Out] It receives the name tag of the font, such as /Helv (get rid of slash).
------------	---

---

**Return**

Non-zero means found, otherwise not found.

**Head file reference**

fpd\_docTempl.h: 4436

**FPDInterFormFindFormFont2****Syntax**

```
FS_BOOL FPDInterFormFindFormFont2 (
    FPD\_InterForm form,
    FS\_LPSTR szFontName,
    FPD\_Font* outFont,
    FS\_ByteString* outNameTag
);
```

**Description**

Finds a font with font name. ANSI version.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

szFontName	[In] The input font name.
------------	---------------------------

---

outFont	[Out] It receives the found PDF font.
---------	---------------------------------------

---



---

outNameTag	[Out] It receives the name tag of the font, such as /Helv (get rid of slash).
------------	---

---

**Return**

Non-zero means found, otherwise not found.

**Head file reference**

fpd\_docTempl.h: 4447

**FPDInterFormFindFormFont3****Syntax**

```
FS_BOOL FPDInterFormFindFormFont3 (
    FPD_InterForm form,
    FS_LPWSTR wszFontName,
    FPD_Font* outFont,
    FS_ByteString* outNameTag
);
```

**Description**

Finds a font with font name. Unicode version.

**Parameter**


---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

wszFontName	[In] The input font name.
-------------	---------------------------

---

outFont	[Out] It receives the found PDF font.
---------	---------------------------------------

---

outNameTag	[Out] It receives the name tag of the font, such as /Helv (get rid of slash).
------------	---

---

**Return**

Non-zero means found, otherwise not found.

**Head file reference**

fpd\_docTempl.h: 4459

**FPDInterFormFixPageFields****Syntax**

```
void FPDInterFormFixPageFields (
    FPD_InterForm form,
    FPD_Page page
);
```

**Description**

Supplement some form field from a page

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

page	[In] The input page dictionary.
------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4715

**FPDInterFormGenerateNewResourceName****Syntax**

```
void FPDInterFormGenerateNewResourceName (
    const FPD_Object resDict,
    FS_LPCSTR csType,
    FS_INT32 iMinLen,
    FS_LPSTR csPrefix,
    FS_BytString* outName
);
```

**Description**

Generate a new resource tag name, especially for ExtGState, ColorSpace, Font, etc.

**Parameter**

---

resDict	[In] An existing resource dictionary.
---------	---------------------------------------

---

csType	[In] Specifies resource type to generate a new name string, currently, it can be ExtGState, ColorSpace, Font, any other is no sense.
--------	--

---

iMinLen	[In] The minimum length of the resource name.
---------	---

---

csPrefix	[In] A prefix string to add in the head of the new resource name.
----------	---

---

outName	[Out] A resource tag name.
---------	----------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3898

**FPDInterForm GetAllFieldNames****Syntax**

```
void FPDInterFormGetAllFieldNames (
    FPD\_InterForm form,
    FS\_WideStringArray* outAllFieldNames
);
```

**Description**

Gets all field names in the form.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

outAllFieldNames	[Out] It receives all field names in the form.
------------------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4076

**FPDInterForm GetControl****Syntax**

```
FPD_FormControl FPDInterFormGetControl (
    FPD\_InterForm form,
    FS\_DWORD index,
    FS\_LPWSTR wszFieldName
);
```

**Description**

Gets a form control.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

---

index	[In] The zero-based control index in the matched control array.
-------	---

---

wszFieldName	[In] The field name to be matched. -empty means for all fields; or system will match the field name. -as shortest string, for example, text1 will match text1.0, text1.2.0, etc. but text1 will not match test10 or test11.1..
--------------	--

---

**Return**

A form control.

**Head file reference**

fpd\_docTempl.h: 4118

**FPDInterFormGetControlAtPoint****Syntax**

```
FPD_FormControl FPDInterFormGetControlAtPoint (
    FPD\_InterForm form,
    FPD\_Page page,
    FS\_FLOAT pdfX,
    FS\_FLOAT pdfY
);
```

**Description**

Retrieves the control by mouse position.

**Parameter**


---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

page	[In] The input PDF page.
------	--------------------------

---

pdfX	[In] The x-coordinate of mouse position in user space.
------	--

---

pdfY	[In] The y-coordinate of mouse position in user space.
------	--

---

**Return**

A control.

**Head file reference**

fpd\_docTempl.h: 4162

**FPDInterFormGetControlByDict****Syntax**

---

```
FPDFormControl FPDInterFormGetControlByDict (
    FPD\_InterForm form,
    FPD\_Object widgetDict
);
```

**Description**

Retrieves the control by widget dictionary.

**Parameter**


---

form	[In] The input PDF interactive form.
widgetDict	[In] The input widget dictionary.

---

**Return**

A control.

**Head file reference**

[fpd\\_docTempl.h](#): 4174

**FPDInterFormGetDefaultAppearance****Syntax**

```
void FPDInterFormGetDefaultAppearance (
    FPD\_InterForm form,
    FPD\_DefaultAppearance* outDefAP
);
```

**Description**

Gets the default appearance interpreter.

**Parameter**


---

form	[In] The input PDF interactive form.
outDefAP	[Out] The default appearance interpreter.

---

**Return**

void

**Head file reference**

[fpd\\_docTempl.h](#): 4523

**Note:**Do not forget to save default appearance after you change any setting in it when you retrieve an object of [FPD\\_DefaultAppearance](#) by GetDefaultAppearance method.

## FPDInterFormGetDefaultFormFont

### Syntax

```
FPD_Font FPDInterFormGetDefaultFormFont (
    FPD\_InterForm form
);
```

### Description

Gets the default form font.

### Parameter

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

### Return

The default form font.

### Head file reference

fpd\_docTempl.h: 4544

## FPDInterFormGetDocument

### Syntax

```
FPD_Document FPDInterFormGetDocument (
    FPD\_InterForm form
);
```

### Description

Gets the PDF document.

### Parameter

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

### Return

The PDF document.

### Head file reference

fpd\_docTempl.h: 4276

## FPDInterFormGetField

### Syntax

```
FPD_FormField FPDInterFormGetField (
    FPD\_InterForm form,
    FS\_DWORD index,
    FS\_LPWSTR wszFieldName
);
```

**Description**

Gets a form field.

**Parameter**

form	[In] The input PDF interactive form.
index	[In] The zero-based field index in the matched field array.
wszFieldName	[In] The field name to be matched. -empty means for all fields; or system will match the field name. -as shortest string, for example, text1 will match text1.0, text1.2.0, etc. but text1 will not match test10 or test11.1..

**Return**

A form field.

**Head file reference**

fpd\_docTempl.h: 4063

**FPDInterFormGetFieldByDict****Syntax**

```
FPD_FormField FPDInterFormGetFieldByDict (  
    FPD\_InterForm form,  
    FPD\_Object fieldDict  
)
```

**Description**

Retrieves the field by field dictionary.

**Parameter**

form	[In] The input PDF interactive form.
fieldDict	[In] The input form field dictionary.

**Return**

A form field.

**Head file reference**

fpd\_docTempl.h: 4096

## FPDInterFormGetFieldInCalculationOrder

### Syntax

```
FPD_FormField FPDInterFormGetFieldInCalculationOrder (
    FPD\_InterForm form,
    FS_INT32 index
);
```

### Description

Gets a form field from the calculation array.

### Parameter

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

index	[In] The zero-based field index in the CO array.
-------	--

---

### Return

A form field.

### Head file reference

fpd\_docTempl.h: 4322

## FPDInterFormGetFormAlignment

### Syntax

```
FS_INT32 FPDInterFormGetFormAlignment (
    FPD\_InterForm form
);
```

### Description

Gets the form alignment.

### Parameter

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

### Return

alignment value:

- 0 - left alignment, the default setting.
- 1 - centered.
- 2 - right alignment.

### Head file reference

fpd\_docTempl.h: 4563

## FPDInterFormGetFormDict

### Syntax

```
FPD_Object FPDInterFormGetFormDict (
    FPD\_InterForm form
);
```

### Description

Gets the AcroForm dictionary.

### Parameter

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

### Return

The AcroForm dictionary.

### Head file reference

fpd\_docTempl.h: 4285

## FPDInterFormGetFormFont

### Syntax

```
FPD_Font FPDInterFormGetFormFont (
    FPD\_InterForm form,
    FS\_DWORD index,
    FS\_ByteString* outNameTag
);
```

### Description

Gets a PDF font from the font resources.

### Parameter

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

---

index	[In] The zero-based font index in the font resources.
-------	---

---

---

outNameTag	[Out] It receives the name tag of the font, such as /Helv (get rid of slash).
------------	---

---

### Return

A PDF font.

### Head file reference

---

fpd\_docTempl.h: 4383

## FPDInterFormGetFormFont2

### Syntax

```
FPD_Font FPDInterFormGetFormFont2 (
    FPD\_InterForm form,
    FS\_LPSTR szNameTag
);
```

### Description

Gets a PDF font with specified name tag.

### Parameter

---

form	[In] The input PDF interactive form.
szNameTag	[In] the name tag of the font, such as /Helv (get rid of slash).

---

### Return

A PDF font.

### Head file reference

fpd\_docTempl.h: 4394

## FPDInterFormGetFormFont3

### Syntax

```
FPD_Font FPDInterFormGetFormFont3 (
    FPD\_InterForm form,
    FS\_LPSTR szFontName,
    FS\_ByteString* outNameTag
);
```

### Description

Finds a PDF font with specified font name.

### Parameter

---

form	[In] The input PDF interactive form.
szFontName	[In] The input font name.

---

outNameTag	[Out] It receives the name tag of the font, such as /Helv (get rid of slash).
------------	---

---

**Return**

A PDF font.

**Head file reference**

fpd\_docTempl.h: 4404

**FPDInterFormGetFormNotify****Syntax**

```
FPD_FormNotify FPDInterFormGetFormNotify (
    FPD\_InterForm form
);
```

**Description**

Gets the form notify handler.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

**Return**

A form notify handler.

**Head file reference**

fpd\_docTempl.h: 3833

**FPDInterFormGetInternalField****Syntax**

```
FPD_Object FPDInterFormGetInternalField (
    FPD\_InterForm form,
    FS\_DWORD index,
    FS\_LPWSTR wszFieldName
);
```

**Description**

Gets an internal field.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

index	[In] The zero-based internal field index in the matched internal field array.
-------	---

---

---

wszFieldName	[In] The field name to be matched. -If empty, this will retrieve all internal field dictionaries from the root list of the whole form. -If has value, all internal fields which names match the csfieldName will return.
--------------	--

---

**Return**

The internal field dictionary.

**Head file reference**

fpd\_docTempl.h: 4263

**FPDInterFormGetNativeFont****Syntax**

```
void FPDInterFormGetNativeFont (
    FS_BYTE iCharSet,
    FS_LPVVOID pLogFont,
    FS_ByteString* outFont
);
```

**Description**

Gets the native font name with a character set and a logical font structure.

**Parameter**


---

iCharSet	[In] The input character set.
----------	-------------------------------

---

pLogFont	[Out] It receives the font information. Points to LOGFONTA structure.
----------	---

---

outFont	[Out] The font name.
---------	----------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3953

**FPDInterFormGetNativeFont2****Syntax**

```
void FPDInterFormGetNativeFont2 (
    FS_LPVVOID pLogFont,
    FS_ByteString* outFont
);
```

**Description**

Gets the native font name with a logical font structure.

**Parameter**

---

pLogFont	[Out] It receives the font information. Points to LOGFONTA structure.
----------	---

---

outFont	[Out] The font name.
---------	----------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3964

**FPDInterFormGetNativeFormFont****Syntax**

```
FPD_Font FPDInterFormGetNativeFormFont (
    FPD_InterForm form,
    FS_BYTE iCharSet,
    FS_ByteString* outNameTag
);
```

**Description**

Gets a native-form font with a character set.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

iCharSet	[In] The input character set.
----------	-------------------------------

---

outNameTag	[Out] It receives the name tag of the font, such as /Helv (get rid of slash).
------------	---

---

**Return**

A PDF font.

**Head file reference**

fpd\_docTempl.h: 4415

**FPDInterFormGetNativeFormFont2****Syntax**

```
FPD_Font FPDInterFormGetNativeFormFont2 (
    FPD\_InterForm form,
    FS\_ByteString* outNameTag
);
```

**Description**

Gets the native-form font.

**Parameter**

---

form	[In] The input PDF interactive form.
outNameTag	[Out] It receives the name tag of the font, such as /Helv (get rid of slash).

---

**Return**

A PDF font.

**Head file reference**

fpd\_docTempl.h: 4426

**FPDInterFormGetPageControl****Syntax**

```
FPD_FormControl FPDInterFormGetPageControl (
    FPD\_InterForm form,
    FPD\_Page page,
    FS\_INT32 index
);
```

**Description**

Gets a control in specified page.

**Parameter**

---

form	[In] The input PDF interactive form.
page	[In] The input PDF page.
index	[In] The zero-based control index in the page.

---

**Return**

A control int the page.

**Head file reference**

fpd\_docTempl.h: 4151

## FPDInterFormGetPageWithWidget

### Syntax

```
FS_INT32 FPDInterFormGetPageWithWidget (
    FPD\_InterForm form,
    FS\_INT32 iCurPage,
    FS\_BOOL bNext
);
```

### Description

Retrieves the next or previous page for the page specified by iCurPage.

### Parameter

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

iCurPage	[In] The current page index.
----------	------------------------------

---

bNext	[In] Whether to retrieve the next or previous page.
-------	---

---

### Return

The next or previous page index.

### Head file reference

fpd\_docTempl.h: 4686

## FPDInterFormImportFromFDF

### Syntax

```
FS_BOOL FPDInterFormImportFromFDF (
    FPD\_InterForm form,
    const FDF Document FDFDoc,
    FS\_BOOL bNotify
);
```

### Description

Imports a FDF document.

### Parameter

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

FDFDoc	[In] The input FDF document.
--------	------------------------------

---

---

bNotify	[In] Whether do notifying.
---------	----------------------------

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 4625

**FPDInterFormInsertFieldInCalculationOrder****Syntax**

```
FS_INT32 FPDInterFormInsertFieldInCalculationOrder (
    FPD\_InterForm form,
    const FPD\_FormField field,
    FS_INT32 index
);
```

**Description**

Inserts a form field to the CO array.

**Parameter**


---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

field	[In] The input form field.
-------	----------------------------

---

index	[In] The zero-based field index in the CO array to insert at.
-------	---

---

**Return**

The zero-based index of the inserted field. -1 means failed.

**Head file reference**

fpd\_docTempl.h: 4342

**FPDInterFormIsUpdated****Syntax**

```
FS_BOOL FPDInterFormIsUpdated (
    FPD\_InterForm form
);
```

**Description**

Checks whether the form is updated.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

**Return**

[TRUE](#) for the form being updated.

**Head file reference**

fpd\_docTempl.h: 4697

**FPDInterFormIsValidFormControl****Syntax**

```
FS_BOOL FPDInterFormIsValidFormControl (
    FPD\_InterForm form,
    const void* pControl
);
```

**Description**

Checks whether a pointer is a form control pointer.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

---

pControl	[In] The input pointer.
----------	-------------------------

---

**Return**

Non-zero means it's a form control pointer, otherwise is not.

**Head file reference**

fpd\_docTempl.h: 4131

**FPDInterFormIsValidFormField****Syntax**

```
FS_BOOL FPDInterFormIsValidFormField (
    FPD\_InterForm form,
    const void* pField
);
```

**Description**

Checks whether a pointer is a form field pointer.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

---

pField	[In] The input pointer.
--------	-------------------------

---

**Return**

Non-zero means it's a form field pointer, otherwise is not.

**Head file reference**

fpd\_docTempl.h: 4086

**FPDInterFormMoveFieldInCalculationOrder****Syntax**

```
FS_INT32 FPDInterFormMoveFieldInCalculationOrder (
    FPD\_InterForm form,
    const FPD\_FormField field,
    FS\_INT32 iNewIndex
);
```

**Description**

Moves a form field in the CO array to another position.

**Parameter**


---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

field	[In] The input form field.
-------	----------------------------

---

iNewIndex	[In] The new zero-based field index in the CO array.
-----------	--

---

**Return**

The current index of specified field. -1 means failed.

**Head file reference**

fpd\_docTempl.h: 4353

**FPDInterFormNeedConstructAP****Syntax**

```
FS_BOOL FPDInterFormNeedConstructAP (
    FPD\_InterForm form
);
```

**Description**

Checks whether to construct appearance streams and appearance dictionaries for all widget annotations in the document.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

**Return**

[TRUE](#) for constructing appearance streams and appearance dictionaries, otherwise not.

**Head file reference**

fpd\_docTempl.h: 4294

**FPDInterFormNeedConstructAP2****Syntax**

```
void FPDInterFormNeedConstructAP2 (
    FPD\_InterForm form,
    FS\_BOOL bNeedAP
);
```

**Description**

Sets the construct flag.

**Parameter**


---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

bNeedAP	[In] The input construct flag.
---------	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4303

**FPDInterFormNew****Syntax**

```
FPD_InterForm FPDInterFormNew (
    FPD\_Document doc,
    FS\_BOOL bUpdateAP
);
```

**Description**

Creates a form from a document. The bUpdateAP param specifies whether we need to regenerate appearance streams for the fields.

**Parameter**


---

doc	[In] The PDF document.
-----	------------------------

---

---

bUpdateAP	[In] Whether we need to regenerate appearance streams for fields.
-----------	---

---

**Return**

A PDF interactive form.

**Head file reference**

fpd\_docTempl.h: 3860

**FPDInterFormNewControl****Syntax**

```
FPD_FormControl FPDInterFormNewControl (
    FPD\_InterForm form,
    FS\_WideString* inOutFieldName,
    FS\_INT32 iType
);
```

**Description**

Creates and add a control in form, param iType uses FPD\_FORM\_FIELDTYPE\_X macros. inOutFieldName is in/out param. If succeed, inOutFieldName is the validated field full name.

**Parameter**


---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

inOutFieldName	[In/Out] Input a field name and receives the valid field full name.
----------------	---

---

iType	[In] The input field type.
-------	----------------------------

---

**Return**

A form control.

**Head file reference**

fpd\_docTempl.h: 4038

**Note:** This Function will call NewField if necessary. Use this Function to create a control and use returned object to initialize Widget.

**FPDInterFormNewField****Syntax**

```
FPD_FormField FPDInterFormNewField (
    FPD\_InterForm form,
    FS\_WideString* inOutFieldName,
```

```
FS_INT32 iType  
);
```

**Description**

Creates and add a field in form, param iType uses FPD\_FORM\_FIELDTYPE\_X macros.  
inOutFieldName is in/out param. If succeed, csFieldName is the validated field full name.

**Parameter**

form	[In] The input PDF interactive form.
inOutFieldName	[In/Out] Input a field name and receives the valid field full name.
iType	[In] The input field type.

**Return**

A form field.

**Head file reference**

fpd\_docTempl.h: 4026

**FPDInterFormReloadForm****Syntax**

```
void FPDInterFormReloadForm (  
    FPD_InterForm form  
);
```

**Description**

Reload the interactive form.

**Parameter**

form	[In] The input PDF interactive form.
------	--------------------------------------

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4658

**FPDInterFormRemoveFieldInCalculationOrder****Syntax**

```
void FPDInterFormRemoveFieldInCalculationOrder (
    FPD_InterForm form,
    const FPD_FormField field
);
```

**Description**

Removes a field in the CO array.

**Parameter**

form	[In] The input PDF interactive form.
field	[In] The input form field to be removed.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4364

**FPDInterFormRemoveFormFont****Syntax**

```
void FPDInterFormRemoveFormFont (
    FPD_InterForm form,
    const FPD_Font font
);
```

**Description**

Removes a form font with font pointer.

**Parameter**

form	[In] The input PDF interactive form.
font	[In] The input PDF font pointer.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4503

**FPDInterFormRemoveFormFont2**

**Syntax**

```
void FPDInterFormRemoveFormFont2 (
    FPD\_InterForm form,
    FS\_LPCSTR szNameTag
);
```

**Description**

Removes a form font with specified name tag.

**Parameter**

form	[In] The input PDF interactive form.
szNameTag	[In] The input name tag of the font, such as /Helv (get rid of slash).

**Return**

void

**Head file reference**

[fpd\\_docTempl.h](#): 4513

**FPDInterFormRenameControl****Syntax**

```
FS_BOOL FPDInterFormRenameControl (
    FPD\_InterForm form,
    FPD\_FormControl pControl,
    FS\_WideString* inOutNewName
);
```

**Description**

Rename a control's full name.

**Parameter**

form	[In] The input PDF interactive form.
pControl	[In] The input control.

  

inOutNewName	[In/Out] Input the control's new name and receive the validated control full name.if success.
--------------	---

**Return**

[TRUE](#) for success, otherwise not.

**Head file reference**

fpd\_docTempl.h: 4206

**FPDInterFormRenameField****Syntax**

```
FS_BOOL FPDInterFormRenameField (
    FPD\_InterForm form,
    FS\_LPWSTR wszOldName,
    FS\_WideString* inOutNewName
);
```

**Description**

Rename a field's full name. The field identified by it's old full name.

**Parameter**

form	[In] The input PDF interactive form.
wszOldName	[In] The field's old name.
inOutNewName	[In/Out] Input the field's new name and receive the validated field full name, if success.

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 4184

**FPDInterFormRenameField2****Syntax**

```
FS_BOOL FPDInterFormRenameField2 (
    FPD\_InterForm form,
    FPD\_FormField field,
    FS\_WideString* inOutNewName
);
```

**Description**

Rename a field's full name.

**Parameter**

form	[In] The input PDF interactive form.
------	--------------------------------------

---

field	[In] The input field.
inOutNewName	[In/Out] Input the field's new name and receive the validated field full name.if success.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 4195

**FPDInterFormResetForm****Syntax**

```
FS_BOOL FPDInterFormResetForm (
    FPD\_InterForm form,
    const FS\_PtrArray fields,
    FS\_BOOL bIncludeOrExclude,
    FS\_BOOL bNotify
);
```

**Description**

Resets form fields.

**Parameter**


---

form	[In] The input PDF interactive form.
fields	[In] The input form fields array. Fields is the array of <a href="#">FPD_FormField</a> .
bIncludeOrExclude	[In] Whether to include or exclude the form fields.
bNotify	[In] Whether to do notifying.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 4636

**FPDInterFormResetForm2****Syntax**

```
FS_BOOL FPDInterFormResetForm2 (
```

```
FPD_InterForm form,  
FS_BOOL bNotify  
);
```

**Description**

Resets all form fields.

**Parameter**

---

form	[In] The input PDF interactive form.
bNotify	[In] Whether to do notifying.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 4648

**FPDInterFormSetDefaultAppearance****Syntax**

```
void FPDInterFormSetDefaultAppearance (  
    FPD_InterForm form,  
    const FPD_DefaultAppearance cDA  
) ;
```

**Description**

Sets the default appearance interpreter.

**Parameter**

---

form	[In] The input PDF interactive form.
cDA	[In] The input default appearance interpreter.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4534

**FPDInterFormSetFont****Syntax**

```
void FPDInterFormSetFont (
```

FPD\_InterForm form,

const FPD\_Font font

```
);
```

**Description**

Sets the default form font.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

font	[In] The input font.
------	----------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 4553

**FPDInterFormSetFormAlignment****Syntax**

```
void FPDInterFormSetFormAlignment (
```

FPD\_InterForm form,

FS\_INT32 iAlignment

```
);
```

**Description**

Sets the form alignment.

**Parameter**

---

form	[In] The input PDF interactive form.
------	--------------------------------------

---

	[In] The input form alignment type.
--	-------------------------------------

iAlignment

- 0 - left alignment, the default setting.
  - 1 - centered,
  - 2 - right alignment.
- 

**Return**

void

**Head file reference**

---

fpd\_docTempl.h: 4577

## FPDInterFormSetFormNotify

### Syntax

```
void FPDInterFormSetFormNotify (
    FPD\_InterForm form,
    pNotify
);
```

### Description

Sets the form notify handler.

### Parameter

form	[In] The input PDF interactive form.
------	--------------------------------------

pNotify	[In] The notify callbacks.
---------	----------------------------

### Return

A previous form notify handler.

### Head file reference

fpd\_docTempl.h: 3834

## FPDInterFormUpdatingAPEnabled

### Syntax

```
FS_BOOL FPDInterFormUpdatingAPEnabled (void );
```

### Description

Checks whether need to update appearance automatically.

### Return

[TRUE](#) for needing to update appearance automatically.

### Head file reference

fpd\_docTempl.h: 3889

## FPDInterFormValidateFieldName

### Syntax

```
FS_BOOL FPDInterFormValidateFieldName (
    FPD\_InterForm form,
    FS\_WideString* inOutNewFieldName,
    FS\_INT32 iType
```

```
 );
```

**Description**

Checks the field name of specified type is valid or not, receives the valid field name when it's invalid.

**Parameter**


---

form	[In] The input PDF interactive form.
inOutNewFieldName	[In/Out] Input a field name and receives the valid field name.
iType	[In] The field type. uses FPD_FORM_FIELDTYPE_X macros.

---

**Return**

Non-zero means the input field name is valid, otherwise invalid.

**Head file reference**

fpd\_docTempl.h: 3993

**FPDInterFormValidateFieldName2****Syntax**

```
FS_BOOL FPDInterFormValidateFieldName2 (
    FPD_InterForm form,
    const FPD_FormField field,
    FS_WideString* inOutNewFieldName
);
```

**Description**

Checks the field name for the input field is valid or not, receives the valid field name when it's invalid.

**Parameter**


---

form	[In] The input PDF interactive form.
field	[In] The input field.
inOutNewFieldName	[In/Out] Input a field name and receives the valid field name.

---

**Return**

Non-zero means the input field name is valid, otherwise invalid.

**Head file reference**

---

fpd\_docTempl.h: 4004

### FPDInterFormValidateFieldName3

#### Syntax

```
FS_BOOL FPDInterFormValidateFieldName3 (
    FPD_InterForm form,
    const FPD_FormControl control,
    FS_WideString* inOutNewFieldName
);
```

#### Description

Checks the field name for the input control is valid or not, receives the valid field name when it's invalid.

#### Parameter

form	[In] The input PDF interactive form.
control	[In] The input control.
inOutNewFieldName	[In/Out] Input a field name and receives the valid field name.

#### Return

Non-zero means the input field name is valid, otherwise invalid.

#### Head file reference

fpd\_docTempl.h: 4015

## FPD\_Link

### [Return from Used by](#)

#### Description

A FPD\_Link corresponds to a link annotation(see Sections 8.4.5, Annotation Types, the PDF Reference). See [FPDLinkNew](#) , [FPDLinkDestroy](#) [FPDLinkGetLinkAtPoint](#) , [FPDLinkGetLink](#) .

#### Returned from

[FPDLinkExtractNew](#)  
[FPDLinkNew](#)

#### Used by

[\*\*FPDLinkExtractCountLinks\*\*](#)  
[\*\*FPDLinkExtractDestroy\*\*](#)  
[\*\*FPDLinkExtractExtractLinks\*\*](#)  
[\*\*FPDLinkExtractGetRects\*\*](#)  
[\*\*FPDLinkExtractGetURL\*\*](#)  
[\*\*FPDLinkDestroy\*\*](#)  
[\*\*FPDLinkGetAction\*\*](#)  
[\*\*FPDLinkGetDest\*\*](#)  
[\*\*FPDLinkGetLink\*\*](#)  
[\*\*FPDLinkGetLinkAtPoint\*\*](#)  
[\*\*FPDLinkGetRect\*\*](#)

## Functions

### Functions summary

#### [\*\*FPDLinkCountLinks\*\*](#)

Gets the count of links in specified page.

#### [\*\*FPDLinkDestroy\*\*](#)

Destroys a PDF link.

#### [\*\*FPDLinkGetAction\*\*](#)

Gets the action of the link.

#### [\*\*FPDLinkGetDest\*\*](#)

Gets the destination of the link.

#### [\*\*FPDLinkGetLink\*\*](#)

Gets a PDF link.

#### [\*\*FPDLinkGetLinkAtPoint\*\*](#)

Gets the link at specified point. The point is specified in user space. [NULL](#) for no link at that point.

#### [\*\*FPDLinkGetRect\*\*](#)

Gets the rectangle in which the link should be activated.

#### [\*\*FPDLinkNew\*\*](#)

Creates a PDF link from a PDF dictionary.

### Functions detail

#### [\*\*FPDLinkCountLinks\*\*](#)

##### **Syntax**

```
FS_INT32 FPDLinkCountLinks (
    FPD Document doc,
    FPD Page page
);
```

##### **Description**

Gets the count of links in specified page.

##### **Parameter**

---

doc

[In] The input PDF document.

---

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

The count of links in the page.

**Head file reference**

fpd\_docTempl.h: 3247

**FPDLinkDestroy****Syntax**

```
void FPDLinkDestroy (
    FPD\_Link link
);
```

**Description**

Destroys a PDF link.

**Parameter**

---

link	[In] The input PDF link.
------	--------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3224

**FPDLinkGetAction****Syntax**

```
void FPDLinkGetAction (
    FPD\_Link link,
    FPD\_Action* outAction
);
```

**Description**

Gets the action of the link.

**Parameter**

---

link	[In] The input PDF link.
------	--------------------------

---

---

outAction	[Out] The action of the link.
-----------	-------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3289

**FPDLinkGetDest****Syntax**

```
void FPDLinkGetDest (
    FPD\_Link link,
    FPD\_Document doc,
    FPD\_Dest* outDest
);
```

**Description**

Gets the destination of the link.

**Parameter**

---

link	[In] The input PDF link.
------	--------------------------

---

doc	[In] The input PDF document.
-----	------------------------------

---

outDest	[Out] The destination of the link.
---------	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3278

**FPDLinkGetLink****Syntax**

```
void FPDLinkGetLink (
    FPD\_Document doc,
    FPD\_Page page,
    FS\_INT32 index,
    FPD\_Link* outLink
);
```

**Description**

Gets a PDF link.

**Parameter**

---

doc	[In] The input PDF document.
page	[In] The input PDF page.
index	[In] The zero-based link index int the page.
outLink	[Out] It receives the link by index.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3233

**FPDLinkGetLinkAtPoint****Syntax**

```
void FPDLinkGetLinkAtPoint (
    FPD Document doc,
    FPD Page page,
    FS FLOAT x,
    FS FLOAT y,
    FPD Link* outLink
);
```

**Description**

Gets the link at specified point. The point is specified in user space. [NULL](#) for no link at that point.

**Parameter**


---

doc	[In] The input PDF document.
page	[In] The input PDF page.
x	[In] The input x-coordinate of the point.
y	[In] The input y-coordinate of the point.
outLink	[Out] It receives the link at specified point.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3233

**FPDLinkGetRect****Syntax**

```
FS_FloatRect FPDLinkGetRect (
    FPD Link link
);
```

**Description**

Gets the rectangle in which the link should be activated.

**Parameter**

---

link	[In] The input link.
------	----------------------

---

**Return**

The rectangle in which the link should be activated.

**Head file reference**

fpd\_docTempl.h: 3269

**FPDLinkNew****Syntax**

```
FPD_Link FPDLinkNew (
    FPD Object dict
);
```

**Description**

Creates a PDF link from a PDF dictionary.

**Parameter**

---

dict	[In] The input PDF dictionary.
------	--------------------------------

---

**Return**

A PDF link.

**Head file reference**

fpd\_docTempl.h: 3215

**FPD\_LinkExtract**

## **Return from Used by**

### **Description**

A link extractor for accessing links. See [FPDLinkExtractNew](#) , [FPDLinkExtractDestroy](#) .

### **Returned from**

#### [\*\*FPDLinkExtractNew\*\*](#)

### **Used by**

[\*\*FPDLinkExtractCountLinks\*\*](#)  
[\*\*FPDLinkExtractDestroy\*\*](#)  
[\*\*FPDLinkExtractExtractLinks\*\*](#)  
[\*\*FPDLinkExtractGetRects\*\*](#)  
[\*\*FPDLinkExtractGetURL\*\*](#)

### **Functions**

#### **Functions summary**

##### [\*\*FPDLinkExtractCountLinks\*\*](#)

Gets the count of links in the text page.

##### [\*\*FPDLinkExtractDestroy\*\*](#)

Destroys the link extracting object.

##### [\*\*FPDLinkExtractExtractLinks\*\*](#)

Extracts links for a text page.

##### [\*\*FPDLinkExtractGetRects\*\*](#)

Gets the rectangle array for specified link.

##### [\*\*FPDLinkExtractGetURL\*\*](#)

Gets the linked URL for specified link.

##### [\*\*FPDLinkExtractNew\*\*](#)

Creates an empty link extracting object.

#### **Functions detail**

##### [\*\*FPDLinkExtractCountLinks\*\*](#)

###### **Syntax**

```
FS_INT32 FPDLinkExtractCountLinks (
    FPD\_LinkExtract extr
);
```

###### **Description**

Gets the count of links in the text page.

###### **Parameter**

---

<b>extr</b>	[In] The input link extracting object.
-------------	--

---

**Return**

The count of links in the text page. Returns -1 for error.

**Head file reference**

fpd\_textTempl.h: 432

**FPDLinkExtractDestroy****Syntax**

```
void FPDLinkExtractDestroy (
    FPD\_LinkExtract extr
);
```

**Description**

Destroys the link extracting object.

**Parameter**

---

extr	[In] The input link extracting object.
------	--

---

**Return**

void

**Head file reference**

fpd\_textTempl.h: 413

**FPDLinkExtractExtractLinks****Syntax**

```
FS_BOOL FPDLinkExtractExtractLinks (
    FPD\_LinkExtract extr,
    const FPD\_TextPage textpage
);
```

**Description**

Extracts links for a text page.

**Parameter**

---

extr	[In] The input link extracting object.
------	--

---

textpage	[In] The input text page.
----------	---------------------------

---

**Return**

[TRUE](#) means success, otherwise not.

**Head file reference**

fpd\_textTempl.h: 422

**FPDLinkExtractGetRects****Syntax**

```
void FPDLinkExtractGetRects (
    FPD\_LinkExtract extr,
    FS\_INT32 index,
    FS\_FloatRectArray* outRects
);
```

**Description**

Gets the rectangle array for specified link.

**Parameter**

---

extr	[In] The input link extracting object.
------	--

---

index	[In] The specified index.
-------	---------------------------

---

outRects	[Out] It receives the rectangle array for specified link.
----------	---

---

**Return**

void

**Head file reference**

fpd\_textTempl.h: 452

**FPDLinkExtractGetURL****Syntax**

```
void FPDLinkExtractGetURL (
    FPD\_LinkExtract extr,
    FS\_INT32 index,
    FS\_WideString* outURL
);
```

**Description**

Gets the linked URL for specified link.

**Parameter**

---

extr	[In] The input link extracting object.
------	--

---

---

index	[In] The specified index.
outURL	[Out] It receives the linked URL for specified link.

---

**Return**

void

**Head file reference**

fpd\_textTempl.h: 441

**FPDLinkExtractNew****Syntax**

FPD\_LinkExtract FPDLinkExtractNew (void );

**Description**

Creates an empty link extracting object.

**Return**

An empty link extracting object.

**Head file reference**

fpd\_textTempl.h: 405

**FPD\_LWinParam****Return from Used by****Description**

A FPD\_LWinParam represents a set of parameter on Windows desktop to launch an application or open or print a document. It is used by launch actions. See [FPDLWinParamNew](#) , [FPDLWinParamDestroy](#) , [FPDACTIONGetWinParam](#) .

**Returned from****FPDLWinParamNew****Used by**

[FPDACTIONGetWinParam](#)  
[FPDLWinParamDestroy](#)  
[FPDLWinParamGetDefaultDirectory](#)  
[FPDLWinParamGetDict](#)  
[FPDLWinParamGetFileName](#)  
[FPDLWinParamGetOperation](#)  
[FPDLWinParamGetParam](#)

[\*\*FPDLWinParamSetDefaultDirectory\*\*](#)  
[\*\*FPDLWinParamSetFileName\*\*](#)  
[\*\*FPDLWinParamSetOperation\*\*](#)  
[\*\*FPDLWinParamSetParam\*\*](#)

## Functions

### Functions summary

#### [\*\*FPDLWinParamDestroy\*\*](#)

Destroys windows launch param.

#### [\*\*FPDLWinParamGetDefaultDirectory\*\*](#)

Gets the default directory in standard DOS syntax.

#### [\*\*FPDLWinParamGetDict\*\*](#)

Gets the windows launch Param dictionary.

#### [\*\*FPDLWinParamGetFileName\*\*](#)

Gets the file name of the the application to be launched or the document to be opened or printed.

#### [\*\*FPDLWinParamGetOperation\*\*](#)

Gets the operation to perform.

#### [\*\*FPDLWinParamGetParam\*\*](#)

Gets the param to be passed to the application designated by the F entry.

#### [\*\*FPDLWinParamNew\*\*](#)

Creates windows launch param from a PDF dictionary.

#### [\*\*FPDLWinParamSetDefaultDirectory\*\*](#)

Sets the default directory.

#### [\*\*FPDLWinParamSetFileName\*\*](#)

Sets the file name in the windows launch param.

#### [\*\*FPDLWinParamSetOperation\*\*](#)

Sets the operation ASCII string.

#### [\*\*FPDLWinParamSetParam\*\*](#)

Sets the application Params.

### Functions detail

#### FPDLWinParamDestroy

##### **Syntax**

```
void FPDLWinParamDestroy (
    FPD\_LWinParam param
);
```

##### **Description**

Destroys windows launch param.

##### **Parameter**

---

param	[In] Windows launch param.
-------	----------------------------

---

##### **Return**

void

**Head file reference**

fpd\_docTempl.h: 1515

**FPDLWinParamGetDefaultDirectory****Syntax**

```
void FPDLWinParamGetDefaultDirectory (
    FPD\_LWinParam param,
    FS\_ByteString* outDirectory
);
```

**Description**

Gets the default directory in standard DOS syntax.

**Parameter**

---

param	[In] Windows launch param.
-------	----------------------------

---

outDirectory	[Out] The default directory in standard DOS syntax.
--------------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1544

**FPDLWinParamGetDict****Syntax**

```
FPD_Object FPDLWinParamGetDict (
    FPD\_LWinParam param
);
```

**Description**

Gets the windows launch Param dictionary.

**Parameter**

---

param	[In] Windows launch param.
-------	----------------------------

---

**Return**

The windows launch param dictionary.

**Head file reference**

fpd\_docTempl.h: 1604

### FPDLWinParamGetFileName

#### Syntax

```
void FPDLWinParamGetFileName (
    FPD_LWinParam param,
    FS_ByteString* outFileName
);
```

#### Description

Gets the file name of the application to be launched or the document to be opened or printed.

#### Parameter

---

param	[In] Windows launch param.
-------	----------------------------

---

outFileName	[Out] It receives the file name
-------------	---------------------------------

---

#### Return

void

#### Head file reference

fpd\_docTempl.h: 1524

### FPDLWinParamGetOperation

#### Syntax

```
void FPDLWinParamGetOperation (
    FPD_LWinParam param,
    FS_ByteString* outOperation
);
```

#### Description

Gets the operation to perform.

#### Parameter

---

param	[In] Windows launch param.
-------	----------------------------

---

outOperation	[Out] It receives the operation to perform.
--------------	---

---

#### Return

void

**Head file reference**  
fpd\_docTempl.h: 1564**FPDLWinParamGetParam****Syntax**

```
void FPDLWinParamGetParam (
    FPD\_LWinParam param,
    FS\_ByteString* outParam
);
```

**Description**

Gets the param to be passed to the application designated by the F entry.

**Parameter**

---

param	[In] Windows launch param.
-------	----------------------------

---

outParam	[Out] It receives the param to be passed to the application designated by the F entry.
----------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1584

**FPDLWinParamNew****Syntax**

```
FPD_LWinParam FPDLWinParamNew (
    FPD\_Object dict
);
```

**Description**

Creates windows launch param from a PDF dictionary.

**Parameter**

---

dict	[In] The input PDF dictionary.
------	--------------------------------

---

**Return**

Windows launch Param.

**Head file reference**

fpd\_docTempl.h: 1506

## FPDLWinParamSetDefaultDirectory

### Syntax

```
void FPDLWinParamSetDefaultDirectory (
    FPD\_LWinParam param,
    FS\_LPSTR szDirectory
);
```

### Description

Sets the default directory.

### Parameter

---

param	[In] Windows launch param.
-------	----------------------------

---

szDirectory	[In] The new default directory.
-------------	---------------------------------

---

### Return

void

### Head file reference

fpd\_docTempl.h: 1554

## FPDLWinParamSetFileName

### Syntax

```
void FPDLWinParamSetFileName (
    FPD\_LWinParam param,
    FS\_LPSTR szFile
);
```

### Description

Sets the file name in the windows launch param.

### Parameter

---

param	[In] Windows launch param.
-------	----------------------------

---

szFile	[In] The new file name.
--------	-------------------------

---

### Return

void

**Head file reference**

fpd\_docTempl.h: 1534

**FPDLWinParamSetOperation****Syntax**

```
void FPDLWinParamSetOperation (
    FPD\_LWinParam param,
    FS\_LPSTR szOperation
);
```

**Description**

Sets the operation ASCII string.

**Parameter**

---

param	[In] Windows launch param.
-------	----------------------------

---

szOperation	[In] The new operation string.
-------------	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1574

**FPDLWinParamSetParam****Syntax**

```
void FPDLWinParamSetParam (
    FPD\_LWinParam param,
    FS\_LPSTR szParam
);
```

**Description**

Sets the application Params.

**Parameter**

---

param	[In] Windows launch param.
-------	----------------------------

---

szParam	[In] The new application params.
---------	----------------------------------

---

**Return**

void

**Head file reference**  
fpd\_docTempl.h: 1594

## FPD\_MediaPlayer

[Return from Used by](#)

### Description

A media player information object. See [FPDMediaNew](#) , [FPDMediaNewFromDict](#) , [FPDMediaDestroy](#) .

### Returned from

[FPDMediaPlayerNew](#)  
[FPDMediaPlayerNewFromDict](#)

### Used by

[FPDRenditionAddMediaPlayer](#)  
[FPDRenditionCountMediaPlayers](#)  
[FPDRenditionGetMediaPlayer](#)  
[FPDRenditionRemoveMediaPlayer](#)  
[FPDMediaPlayerDestroy](#)  
[FPDMediaPlayerGetOSArray](#)  
[FPDMediaPlayerGetSoftwareURI](#)  
[FPDMediaPlayerSetOSArray](#)  
[FPDMediaPlayerSetSoftwareURI](#)

## Enumerations

### Enumerations summary

#### [FPD\\_MediaPermission](#)

PDF media permission type enumeration.

#### [FPD\\_MediaPlayerType](#)

Enumeration of PDF media player classified type in PDF Media Players dictionary. See [FPDRenditionCountMediaPlayers](#) , [FPDRenditionGetMediaPlayer](#) , [FPDRenditionAddMediaPlayer](#) , [FPDRenditionRemoveMediaPlayer](#) .

#### [FPD\\_MediaPlayParamType](#)

PDF media play parameter type enumeration. See [FPDRenditionSetVolumn](#) .

## Enumerations detail

### FPD\_MediaPermission

#### Syntax

```
enum FPD_MediaPermission{  
    TempNever,  
    TempExtract,
```

```
    TempAccess,  
    TempAlways  
};
```

#### Description

PDF media permission type enumeration. Indicates the circumstances under which it is acceptable to write a temporary file in order to play a media clip. See [FPDRenditionGetPermission](#) .

#### Head file reference

fpd\_docExpT.h: 1215

##### **TempNever**

Never allowed.

##### **TempExtract**

Allowed only if the document permissions allow content extraction.

##### **TempAccess**

Allowed only if the document permissions allow content extraction, including for accessibility purposes.

##### **TempAlways**

Always allowed.

## FPD\_MediaPlayerType

#### Syntax

```
enum FPD_MediaPlayerType{  
    MustUsed,  
    Available,  
    NotUsed  
};
```

#### Description

Enumeration of PDF media player classified type in PDF Media Players dictionary. See [FPDRenditionCountMediaPlayers](#) , [FPDRenditionGetMediaPlayer](#) , [FPDRenditionAddMediaPlayer](#) , [FPDRenditionRemoveMediaPlayer](#) .

#### Head file reference

fpd\_docExpT.h: 1247

##### **MustUsed**

Which must be used in playing the associated media object.

##### **Available**

Which may be used in playing the associated media object.

##### **NotUsed**

Which must not be used in playing the associated media object.

## FPD\_MediaPlayParamType

### Syntax

```
enum FPD_MediaPlayParamType{
    MustBeHonored,
    BestEffort
};
```

### Description

PDF media play parameter type enumeration. See [FPDRenditionSetVolumn](#).

### Head file reference

fpd\_docExpT.h: 1232

### [MustBeHonored](#)

Must be honored for the media play parameters to be considered viable.

### [BestEffort](#)

Need only be honored in a "best effort" sense.

## Functions

### Functions summary

#### [FPDMediaPlayerDestroy](#)

Destroys the media player information object.

#### [FPDMediaPlayerGetOSArray](#)

Gets the OS array in the software identifier dictionary.

#### [FPDMediaPlayerGetSoftwareURI](#)

Gets the software URI.

#### [FPDMediaPlayerNew](#)

Creates an empty media player information object.

#### [FPDMediaPlayerNewFromDict](#)

Creates an media player information object from a PDF dictionary.

#### [FPDMediaPlayerSetOSArray](#)

Sets the OS array.

#### [FPDMediaPlayerSetSoftwareURI](#)

Sets the software URI.

### Functions detail

#### FPDMediaPlayerDestroy

##### Syntax

```
void FPDMediaPlayerDestroy (
    FPD\_MediaPlayer player
);
```

##### Description

Destroys the media player information object.

**Parameter**

---

player	[In] The media player information object.
--------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2561

**FPDMediaPlayerGetOSArray****Syntax**

```
FS_INT32 FPDMediaPlayerGetOSArray (
    FPD_MediaPlayer player,
    FS_ByteStringArray* outOSArray
);
```

**Description**

Gets the OS array in the software identifier dictionary.

**Parameter**

---

player	[In] The media player information object.
--------	---

---

---

outOSArray	[Out] It receives the OS array.
------------	---------------------------------

---

**Return**

The number of OS in the array.

**Head file reference**

fpd\_docTempl.h: 2590

**FPDMediaPlayerGetSoftwareURI****Syntax**

```
void FPDMediaPlayerGetSoftwareURI (
    FPD_MediaPlayer player,
    FS_ByteString* outURL
);
```

**Description**

Gets the software URI.

**Parameter**

---

player	[In] The media player information object.
--------	---

---

outURL	[Out] The software URI.
--------	-------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2570

**FPDMediaPlayerNew****Syntax**

FPD\_MediaPlayer FPDMediaPlayerNew (void );

**Description**

Creates an empty media player information object.

**Return**

An empty media player information object.

**Head file reference**

fpd\_docTempl.h: 2543

**FPDMediaPlayerNewFromDict****Syntax**

FPD\_MediaPlayer FPDMediaPlayerNewFromDict (  
    [FPD\\_Object](#) dict  
);

**Description**

Creates an media player information object from a PDF dictionary.

**Parameter**

---

dict	[In] The input PDF dictionary.
------	--------------------------------

---

**Return**

An media player information object.

**Head file reference**

fpd\_docTempl.h: 2552

**FPDMediaPlayerSetOSArray**

**Syntax**

```
void FPDMediaPlayerSetOSArray (
    FPD\_MediaPlayer player,
    const FS\_ByteStringArray osArray
);
```

**Description**

Sets the OS array.

**Parameter**

---

player	[In] The media player information object.
--------	---

---

osArray	[In] The input OS array.
---------	--------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2600

**FPDMediaPlayerSetSoftwareURI****Syntax**

```
void FPDMediaPlayerSetSoftwareURI (
    FPD\_MediaPlayer player,
    FS\_LPCSTR szURI
);
```

**Description**

Sets the software URI.

**Parameter**

---

player	[In] The media player information object.
--------	---

---

szURI	[In] The input software URI.
-------	------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2580

**FPD\_MeshStream**

## [Return from Used by](#)

### Description

No document exists. See [FPDMeshStreamNew](#) , [FPDMeshStreamDestroy](#) .

### Returned from

#### [FPDMeshStreamNew](#)

### Used by

[FPDMeshStreamDestroy](#)  
[FPDMeshStreamGetColor](#)  
[FPDMeshStreamGetCoords](#)  
[FPDMeshStreamGetFlag](#)  
[FPDMeshStreamGetVertex](#)  
[FPDMeshStreamGetVertexRow](#)

## Functions

### Functions summary

#### [FPDMeshStreamDestroy](#)

Destroys the PDF mesh stream.

#### [FPDMeshStreamGetColor](#)

Reads a vertex color from mesh stream.

#### [FPDMeshStreamGetCoords](#)

Reads a vertex coords from mesh stream.

#### [FPDMeshStreamGetFlag](#)

Reads a vertex flag from mesh stream.

#### [FPDMeshStreamGetVertex](#)

Reads a vertex from mesh stream.

#### [FPDMeshStreamGetVertexRow](#)

Reads a vertex array from mesh stream.

#### [FPDMeshStreamNew](#)

Creates a new PDF mesh stream with the shading stream, functions, and color-spaces.

### Functions detail

#### FPDMeshStreamDestroy

##### Syntax

```
void FPDMeshStreamDestroy (
    FPD_MeshStream meshStream
);
```

##### Description

Destroys the PDF mesh stream.

##### Parameter

---

meshStream	[In] The input PDF mesh stream.
------------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 1897

**FPDMeshStreamGetColor****Syntax**

```
void FPDMeshStreamGetColor (
    FPD\_MeshStream meshStream,
    FS\_FLOAT* outR,
    FS\_FLOAT* outG,
    FS\_FLOAT* outB
);
```

**Description**

Reads a vertex color from mesh stream.

**Parameter**


---

meshStream	[In] The input PDF mesh stream.
------------	---------------------------------

---

outR	[Out] It receives the R value of a vertex color.
------	--

---

outG	[Out] It receives the G value of a vertex color.
------	--

---

outB	[Out] It receives the B value of a vertex color.
------	--

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 1926

**FPDMeshStreamGetCoords****Syntax**

```
void FPDMeshStreamGetCoords (
    FPD\_MeshStream meshStream,
    FS\_FLOAT* outX,
    FS\_FLOAT* outY
);
```

**Description**

Reads a vertex coords from mesh stream.

**Parameter**

---

meshStream	[In] The input PDF mesh stream.
------------	---------------------------------

---

outX	[Out] It receives the X coordinate.
------	-------------------------------------

---

outY	[Out] It receives the Y coordinate.
------	-------------------------------------

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 1915

**FPDMeshStreamGetFlag****Syntax**

```
FS_DWORD FPDMeshStreamGetFlag (
    FPD\_MeshStream meshStream
);
```

**Description**

Reads a vertex flag from mesh stream.

**Parameter**

---

meshStream	[In] The input PDF mesh stream.
------------	---------------------------------

---

**Return**

The vertex flag from mesh stream.

**Head file reference**

fpd\_resourceTempl.h: 1906

**FPDMeshStreamGetVertex****Syntax**

```
FS_DWORD FPDMeshStreamGetVertex (
    FPD\_MeshStream meshStream,
    FPD\_MeshVertex* outVertex,
    FS\_AffineMatrix object2Bitmap
);
```

**Description**

Reads a vertex from mesh stream.

**Parameter**


---

meshStream	[In] The input PDF mesh stream.
outVertex	[Out] It receives the vertex from mesh stream.
object2Bitmap	[In] The input matrix.

---

**Return**

A vertex of *meshStream* .

**Head file reference**

fpd\_resourceTempl.h: 1938

**FPDMeshStreamGetVertexRow****Syntax**

```
FS_BOOL FPDMeshStreamGetVertexRow (
    FPD\_MeshStream meshStream,
    FPD\_MeshVertex* outVertex,
    FS_INT32 count,
    FS\_AffineMatrix object2Bitmap
);
```

**Description**

Reads a vertex array from mesh stream.

**Parameter**


---

meshStream	[In] The input PDF mesh stream.
outVertex	[Out] It receives the vertex from mesh stream.
count	[In] The count of the vertex array.
object2Bitmap	[In] The input matrix.

---

**Return**

[TRUE](#) means success, otherwise [FALSE](#) .

**Head file reference**

fpd\_resourceTempl.h: 1949

**FPDMeshStreamNew****Syntax**

```
FPD_MeshStream FPDMeshStreamNew (
    FPD Object shadingStream,
    FPD Function* pFuncs,
    FS\_INT32 nFuncs,
    FPD ColorSpace cs
);
```

**Description**

Creates a new PDF mesh stream with the shading stream, functions, and color-spaces.

**Parameter**

shadingStream	[In] The input shading stream.
---------------	--------------------------------

pFuncs	[In] The array of 1-in, 1-out functions for shading.
--------	--

nFuncs	[In] The count of functions.
--------	------------------------------

cs	[In] The base color-space for shading.
----	--

**Return**

A new PDF mesh stream.

**Head file reference**

fpd\_resourceTempl.h: 1885

**FPD\_Name**[Return from Used by](#)**Description**

A FPD\_Name is a sequence of non-white space characters. In code, a name is preceded by the forward slash (/) character indicating that it is a string literal, for example: /AName. See Section 3.2.4 in the *PDF Reference* for details.

**Returned from**

[FPDNameTreeNew](#)  
[FPDNameTreeNew2](#)

### Used by

[FPDNameTreeDestroy](#)  
[FPDNameTreeGetCount](#)  
[FPDNameTreeGetIndex](#)  
[FPDNameTreeGetRoot](#)  
[FPDNameTreeLookupNamedDest](#)  
[FPDNameTreeLookupValue](#)  
[FPDNameTreeLookupValueByName](#)  
[FPDNameTreeRemove](#)  
[FPDNameTreeSetValue](#)

## Functions

### Functions summary

#### [FPDNameGetString](#)

Gets a ref to the data of the name object.

#### [FPDNameIdentical](#)

Compares with another name object.

#### [FPDNameNew](#)

Creates a name object from a byte string.

### Functions detail

#### FPDNameGetString

##### Syntax

```
void FPDNameGetString (  
    FPD\_Object objName,  
    FS\_ByteString* outString  
)
```

##### Description

Gets a ref to the data of the name object.

##### Parameter

---

objName	[In] The input name object.
---------	-----------------------------

---

outString	[Out] It receives the data of the name object.
-----------	--

---

##### Return

void

##### Head file reference

fpd\_objsTempl.h: 497

## FPDNameIdentical

### Syntax

```
FS_BOOL FPDNameIdentical (
    FPD Object objName,
    FPD Object other_objName
);
```

### Description

Compares with another name object.

### Parameter

---

objName	[In] The input name object.
---------	-----------------------------

---

other_objName	[In] The other name object.
---------------	-----------------------------

---

### Return

Non-zero means identical, otherwise not identical.

### Head file reference

[fpd\\_objsTempl.h](#): 507

## FPDNameNew

### Syntax

```
FPD_Object FPDNameNew (
    FS LPCSTR str
);
```

### Description

Creates a name object from a byte string.

### Parameter

---

str	[In] The input byte string.
-----	-----------------------------

---

### Return

A name object.

### Head file reference

[fpd\\_objsTempl.h](#): 487

### Related method

[FPDOBJECTDESTROY](#)

# FPD\_NameTree

[Return from Used by](#)

## Description

The dictionary used to store all of the Named Destinations in a PDF file. A name tree is used to map FS ByteString to FPD Objects. You create a FPD\_NameTree d locate it where you think is appropriate (perhaps under a page, but most often right under the catalog).

See [FPDNameTreeNew](#) , [FPDNameTreeNew2](#) , [FPDNameTreeDestroy](#) .

## Returned from

[FPDNameTreeNew](#)

[FPDNameTreeNew2](#)

## Used by

[FPDNameTreeDestroy](#)

[FPDNameTreeGetCount](#)

[FPDNameTreeGetIndex](#)

[FPDNameTreeGetRoot](#)

[FPDNameTreeLookupNamedDest](#)

[FPDNameTreeLookupValue](#)

[FPDNameTreeLookupValueByName](#)

[FPDNameTreeRemove](#)

[FPDNameTreeSetValue](#)

## Functions

### Functions summary

[FPDNameTreeDestroy](#)

Destroys a name tree object created by FPDNameTreeNew or FPDNameTreeNew2.

[FPDNameTreeGetCount](#)

Gets the number of key-value pairs in name tree.

[FPDNameTreeGetIndex](#)

Gets index of the name in name tree.

[FPDNameTreeGetRoot](#)

Gets the root dictionary of the tree.

[FPDNameTreeLookupNamedDest](#)

Lookup PDF name destination.

[FPDNameTreeLookupValue](#)

Lookup a PDF name tree node by index that contains the name, and return specified entry value.

[FPDNameTreeLookupValueByName](#)

Lookup a PDF name tree node that contains the name, and return specified entry value.

[FPDNameTreeNew](#)

Creates a name tree object from a root dictionary.

[FPDNameTreeNew2](#)



Creates a name tree object from a root dictionary.

**FPDNameTreeRemove**

Removes the name and entry value by nIndex in name tree, if csName is empty. otherwise ignore nIndex.

**FPDNameTreeSetValue**

Sets the entry value of specified PDF name tree node that contains specified name.

## Functions detail

### FPDNameTreeDestroy

**Syntax**

```
void FPDNameTreeDestroy (
    FPD_NameTree nameTree
);
```

**Description**

Destroys a name tree object created by FPDNameTreeNew or FPDNameTreeNew2.

**Parameter**

---

nameTree	[In] A name tree object created by FPDNameTreeNew or FPDNameTreeNew2.
----------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 736

### FPDNameTreeGetCount

**Syntax**

```
FS_INT32 FPDNameTreeGetCount (
    FPD_NameTree nameTree
);
```

**Description**

Gets the number of key-value pairs in name tree.

**Parameter**

---

nameTree	[In] A name tree object.
----------	--------------------------

---

**Return**

The number of key-value pairs.

**Head file reference**



fpd\_docTempl.h: 811

## FPDNameTreeGetIndex

### Syntax

```
FS_INT32 FPDNameTreeGetIndex (
    FPD\_NameTree nameTree,
    FS\_LPCWSTR wszName
);
```

### Description

Gets index of the name in name tree.

### Parameter

nameTree	[In] A name tree object.
----------	--------------------------

wszName	[In] The name to be searched.
---------	-------------------------------

### Return

The index of the name If found csName. otherwise return -1.

### Head file reference

fpd\_docTempl.h: 790

## FPDNameTreeGetRoot

### Syntax

```
FPD_Object FPDNameTreeGetRoot (
    FPD\_NameTree nameTree
);
```

### Description

Gets the root dictionary of the tree.

### Parameter

nameTree	[In] A name tree object.
----------	--------------------------

### Return

The root dictionary of the tree.

### Head file reference

fpd\_docTempl.h: 820



**FPDNameTreeLookupNamedDest****Syntax**

```
FPD_Object FPDNameTreeLookupNamedDest (
    FPD\_NameTree nameTree,
    FPD\_Document doc,
    FS\_LPSTR szName
);
```

**Description**

Lookup PDF name destination.

**Parameter**

nameTree	[In] A name tree object.
----------	--------------------------

doc	[In] The document.
-----	--------------------

szName	[In] The input name.
--------	----------------------

**Return**

The corresponding destination.

**Head file reference**

[fpd\\_docTempl.h](#): 766

**FPDNameTreeLookupValue****Syntax**

```
FPD_Object FPDNameTreeLookupValue (
    FPD\_NameTree nameTree,
    FS\_INT32 index,
    FS\_WideString* outName
);
```

**Description**

Lookup a PDF name tree node by index that contains the name, and return specified entry value.

**Parameter**

nameTree	[In] A name tree object.
----------	--------------------------

index	[In] The zero-based index of entry value.
-------	---

outName	[Out] The name to be searched.
---------	--------------------------------



**Return**

The value of specified entry of the found PDF name tree node.

**Head file reference**

fpd\_docTempl.h: 745

**FPDNameTreeLookupValueByName****Syntax**

```
FPD_Object FPDNameTreeLookupValueByName (
    FPD\_NameTree nameTree,
    FS\_LPCWSTR wszName
);
```

**Description**

Lookup a PDF name tree node that contains the name, and return specified entry value.

**Parameter**

---

nameTree	[In] A name tree object.
----------	--------------------------

---

wszName	[In] The name to be searched.
---------	-------------------------------

---

**Return**

The value of specified entry of the found PDF name tree node.

**Head file reference**

fpd\_docTempl.h: 756

**FPDNameTreeNew****Syntax**

```
FPD_NameTree FPDNameTreeNew (
    FPD\_Object rootDict
);
```

**Description**

Creates a name tree object from a root dictionary.

**Parameter**

---

rootDict	[In] The root dictionary for the name tree (See Ref Table 3.28).
----------	--

---

**Return**

A name tree object from a root dictionary.

**Head file reference**

fpd\_docTempl.h: 717

**FPDNameTreeNew2****Syntax**

```
FPD_NameTree FPDNameTreeNew2 (
    FPD Document doc,
    FS LPCSTR szCategory
);
```

**Description**

Creates a name tree object from a root dictionary.

**Parameter**

doc	[In] The PDF document.
-----	------------------------

szCategory	[In] The category key name for name tree, like "Dests", "AP", etc. (See Ref Table 3.28).
------------	--

**Return**

A name tree object from a root dictionary.

**Head file reference**

fpd\_docTempl.h: 726

**FPDNameTreeRemove****Syntax**

```
FS_BOOL FPDNameTreeRemove (
    FPD NameTree nameTree,
    FS INT32 index,
    FS LPCWSTR wszName
);
```

**Description**

Removes the name and entry value by nIndex in name tree, if csName is empty. otherwise ignore nIndex.

**Parameter**

nameTree	[In] A name tree object.
----------	--------------------------

index	[In] The zero-based index of entry value.
-------	---

---

wszName	[In] The name to be searched.
---------	-------------------------------

---

**Return**

The value nonzero if successful, otherwise 0.

**Head file reference**

fpd\_docTempl.h: 800

**FPDNameTreeSetValue****Syntax**

```
FS_INT32 FPDNameTreeSetValue (
    FPD_NameTree nameTree,
    FPD_Document doc,
    FS_LPCSTR szKey,
    FS_LPCWSTR wszName,
    FPD_Object value
);
```

**Description**

Sets the entry value of specified PDF name tree node that contains specified name.

**Parameter**


---

nameTree	[In] A name tree object.
----------	--------------------------

---

doc	[In] The document.
-----	--------------------

---

szKey	[In] The key is entry in the names dictionary.
-------	--

---

wszName	[In] The name to be searched.
---------	-------------------------------

---

value	[In] The input entry value.
-------	-----------------------------

---

**Return**

The index of the set value

**Head file reference**

fpd\_docTempl.h: 777

**FPD\_Null****Description**

There is only one [NULL](#) object, which is used to fill empty or uninitialized positions in arrays or dictionaries. See Section 3.2.8 in the PDF Reference for details.

## Functions

### Functions summary

#### [FPDNullNew](#)

Creates a null object.

### Functions detail

#### FPDNullNew

##### Syntax

```
FPD_Object FPDNullNew (void );
```

##### Description

Creates a null object.

##### Return

A null object.

##### Head file reference

fpd\_objsTempl.h: 1445

## FPD\_Number

### Description

FPD\_Number may be specified by signed or unsigned constants. *See Section 3.2.2 in the PDF Reference* for details.

## Functions

### Functions summary

#### [FPDNumberGetFloat](#)

Gets the floating-point value.

#### [FPDNumberGetInteger](#)

Gets the integer value.

#### [FPDNumberGetNumber](#)

Gets the FIX24.8 value.

#### [FPDNumberGetString](#)

Gets a byte string from this object.

#### [FPDNumberIdentical](#)

Compares with another number object.

#### [FPDNumberIsInteger](#)

Whether the number is an integer.



**[FPDNumberNewByFloat](#)**

Creates a number object from a floating-point value.

**[FPDNumberNewByInt](#)**

Creates a number object from an integer.

**[FPDNumberNewByStr](#)**

Creates a number object from a non-buffered byte string.

**[FPDNumberNewFromData](#)**

Creates a number object from data.

**[FPDNumberSetNumber](#)**

Sets a FIX24.8 value.

**[FPDNumberSetString](#)**

Sets a non-buffered byte string.

## Functions detail

### **FPDNumberGetFloat**

**Syntax**

```
FS_FLOAT FPDNumberGetFloat (
    FPD Object objNumber
);
```

**Description**

Gets the floating-point value.

**Parameter**

---

objNumber	[In] The input number object.
-----------	-------------------------------

---

**Return**

The floating-point value.

**Head file reference**

fpd\_objsTempl.h: 402

### **FPDNumberGetInteger**

**Syntax**

```
FS_INT32 FPDNumberGetInteger (
    FPD Object objNumber
);
```

**Description**

Gets the integer value.

**Parameter**

---

objNumber	[In] The input number object.
-----------	-------------------------------

---

**Return**

The integer value.

**Head file reference**

fpd\_objsTempl.h: 374

**FPDNumberGetNumber****Syntax**

```
FS_FLOAT FPDNumberGetNumber (
    FPD Object objNumber
);
```

**Description**

Gets the FIX24.8 value.

**Parameter**

---

objNumber	[In] The input number object.
-----------	-------------------------------

---

**Return**

The FIX24.8 value.

**Head file reference**

fpd\_objsTempl.h: 383

**FPDNumberGetString****Syntax**

```
void FPDNumberGetString (
    FPD Object objNumber,
    FS ByteString* outResult
);
```

**Description**

Gets a byte string from this object.

**Parameter**

---

objNumber	[In] The input number object.
-----------	-------------------------------

---

outResult	[Out] A byte string from this object.
-----------	---------------------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 345

**FPDNumberIdentical****Syntax**

```
FS_BOOL FPDNumberIdentical (
    FPD Object objNumber,
    FPD Object other_number
);
```

**Description**

Compares with another number object.

**Parameter**

---

objNumber	[In] The input number object.
-----------	-------------------------------

---

other_number	[In] The other number object.
--------------	-------------------------------

---

**Return**

Non-zero means identical, otherwise not identical.

**Head file reference**

fpd\_objsTempl.h: 335

**FPDNumberIsInteger****Syntax**

```
FS_BOOL FPDNumberIsInteger (
    FPD Object objNumber
);
```

**Description**

Whether the number is an integer.

**Parameter**

---

objNumber	[In] The input number object.
-----------	-------------------------------

---

**Return**

[TRUE](#) for the number being an integer.

**Head file reference**

fpd\_objsTempl.h: 365

## FPDNumberNewByFloat

### Syntax

```
FPD_Object FPDNumberNewByFloat (
    FS\_FLOAT value
);
```

### Description

Creates a number object from a floating-point value.

### Parameter

---

value	[In] The input floating-point.
-------	--------------------------------

---

### Return

A number object.

### Head file reference

fpd\_objsTempl.h: 288

### Related method

[FPDOBJECTDESTROY](#)  
[FPDNUMBERNEWBYINT](#)  
[FPDNUMBERNEWBYSTR](#)  
[FPDNUMBERNEWFROMDATA](#)

## FPDNumberNewByInt

### Syntax

```
FPD_Object FPDNumberNewByInt (
    FS\_INT32 value
);
```

### Description

Creates a number object from an integer.

### Parameter

---

value	[In] The input integer.
-------	-------------------------

---

### Return

A number object.

### Head file reference

fpd\_objsTempl.h: 282

### Related method

[FPDOBJECTDESTROY](#)



[FPDNumberNewByFloat](#)  
[FPDNumberNewByStr](#)  
[FPDNumberNewFromData](#)

## FPDNumberNewByStr

### Syntax

```
FPD_Object FPDNumberNewByStr (
    FS\_LPCSTR pStr
);
```

### Description

Creates a number object from a non-buffered byte string.

### Parameter

---

pStr	[In] The input non-buffered byte string.
------	--

---

### Return

A number object.

### Head file reference

fpd\_objsTempl.h: 289

### Related method

[FPDOObjectDestroy](#)  
[FPDNumberNewByFloat](#)  
[FPDNumberNewByInt](#)  
[FPDNumberNewFromData](#)

## FPDNumberNewFromData

### Syntax

```
FPD_Object FPDNumberNewFromData (
    FS\_BOOL bInteger,
    void* pData
);
```

### Description

Creates a number object from data.

### Parameter

---

bInteger	[In] Whether the input data is actually an integer.
----------	---

---

pData	[In] The input data.
-------	----------------------

---



**Return**

A number object.

**Head file reference**

fpd\_objsTempl.h: 290

**Related method**

[FPDOObjectDestroy](#)

[FPDNumberNewByFloat](#)

[FPDNumberNewByStr](#)

[FPDNumberNewByInt](#)

## FPDNumberSetNumber

**Syntax**

```
void FPDNumberSetNumber (
    FPD Object objNumber,
    FS FLOAT value
);
```

**Description**

Sets a FIX24.8 value.

**Parameter**

---

objNumber	[In] The input number object.
-----------	-------------------------------

---

value	[In] The input FIX24.8 value.
-------	-------------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 392

## FPDNumberSetString

**Syntax**

```
void FPDNumberSetString (
    FPD Object objNumber,
    FS LPSTR str
);
```

**Description**

Sets a non-buffered byte string.

**Parameter**

---

objNumber	[In] The input number object.
str	[In] The input non-buffered byte string.

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 355

## FPD\_ObjArchiveLoader

**[Return from Used by](#)**

### Description

PDF object archive loader class. See [CFPDObjArchiveLoaderNew](#) , [CFPDObjArchiveLoaderDestroy](#) .

### Returned from

**[FPDObjArchiveLoaderNew](#)**

### Used by

[FPDObjArchiveLoaderDestroy](#)  
[FPDObjArchiveLoaderIsEOF](#)  
[FPDObjArchiveLoaderLoadByte](#)  
[FPDObjArchiveLoaderLoadByteString](#)  
[FPDObjArchiveLoaderLoadDWORD](#)  
[FPDObjArchiveLoaderLoadFloat](#)  
[FPDObjArchiveLoaderLoadInteger](#)  
[FPDObjArchiveLoaderLoadObject](#)  
[FPDObjArchiveLoaderLoadWideString](#)  
[FPDObjArchiveLoaderRead](#)

### Functions

#### Functions summary

**[FPDObjArchiveLoaderDestroy](#)**

Destroys the PDF object archive loader.

**[FPDObjArchiveLoaderIsEOF](#)**

Returns whether de-serializing to the end of the loading buffer.

**[FPDObjArchiveLoaderLoadByte](#)**

Loads a byte value from archive.

**[FPDObjArchiveLoaderLoadByteString](#)**

Loads a byte string value from archive.

**[FPDObjArchiveLoaderLoadDWORD](#)**

Loads a DWORD value from archive.

**[FPDOBJArchiveLoaderLoadFloat](#)**

Loads a float value from archive.

**[FPDOBJArchiveLoaderLoadInteger](#)**

Loads a integer value from archive.

**[FPDOBJArchiveLoaderLoadObject](#)**

Loads an object from archive.

**[FPDOBJArchiveLoaderLoadWideString](#)**

Loads a wide string value from archive.

**[FPDOBJArchiveLoaderNew](#)**

Creates a PDF object archive loader.

**[FPDOBJArchiveLoaderRead](#)**

De-serializes a memory block.

## Functions detail

### **FPDOBJArchiveLoaderDestroy**

**Syntax**

```
void FPDOBJArchiveLoaderDestroy (
    FPD\_ObjArchiveLoader ar
);
```

**Description**

Destroys the PDF object archive loader.

**Parameter**

---

ar	[In] The input object archive loader.
----	---------------------------------------

---

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 173

### **FPDOBJArchiveLoaderIsEOF**

**Syntax**

```
FS_BOOL FPDOBJArchiveLoaderIsEOF (
    FPD\_ObjArchiveLoader ar
);
```

**Description**

Returns whether de-serializing to the end of the loading buffer.

**Parameter**

---

ar	[In] The input object archive loader.
----	---------------------------------------

---

**Return**

non-zero means de-serializing to the end, otherwise not.

**Head file reference**

fpd\_serialTempl.h: 192

**FPDObjArchiveLoaderLoadByte****Syntax**

```
void FPDObjArchiveLoaderLoadByte (
    FPD\_ObjArchiveLoader ar,
    FS\_BYTE* i
);
```

**Description**

Loads a byte value from archive.

**Parameter**


---

ar	[In] Ref to the input object archive loader.
----	--

---

i	[Out] It receives the byte value.
---	-----------------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 212

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDObjArchiveLoaderLoadByteString****Syntax**

```
void FPDObjArchiveLoaderLoadByteString (
    FPD\_ObjArchiveLoader ar,
    FS\_ByteString* pStr
);
```

**Description**

Loads a byte string value from archive.

**Parameter**

---

ar	[In] Ref to the input object archive loader.
----	--

---

pStr	[Out] It receives the byte string value.
------	--

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 256

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FPDObjArchiveLoaderLoadDWORD****Syntax**

```
void FPDObjArchiveLoaderLoadDWORD (
    FPD\_ObjArchiveLoader ar,
    FS\_DWORD* i
);
```

**Description**

Loads a DWORD value from archive.

**Parameter**


---

ar	[In] Ref to the input object archive loader.
----	--

---

i	[Out] It receives the DWORD value.
---	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 234

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FPDObjArchiveLoaderLoadFloat****Syntax**

```
void FPDObjArchiveLoaderLoadFloat (
    FPD\_ObjArchiveLoader ar,
    FS\_FLOAT* i
);
```

**Description**

Loads a float value from archive.

**Parameter**

---

ar	[In] Ref to the input object archive loader.
----	--

---

i	[Out] It receives the float value.
---	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 245

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FPDObjArchiveLoaderLoadInteger****Syntax**

```
void FPDObjArchiveLoaderLoadInteger (
    FPD\_ObjArchiveLoader ar,
    FS\_INT32* i
);
```

**Description**

Loads a integer value from archive.

**Parameter**

---

ar	[In] Ref to the input object archive loader.
----	--

---

i	[Out] It receives the integer value.
---	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 223

**Since**

[SDK LATEEST VERSION > 1.0](#)

## FPDOBJArchiveLoaderLoadObject

### Syntax

```
void FPDOBJArchiveLoaderLoadObject (
    FPD\_ObjArchiveLoader ar,
    FPD\_Object* pObj
);
```

### Description

Loads an object from archive.

### Parameter

---

ar	[In] Ref to the input object archive loader.
----	--

---

pObj	[Out] It receives the loaded PDF object.
------	--

---

### Return

void

### Head file reference

fpd\_serialTempl.h: 182

## FPDOBJArchiveLoaderLoadWideString

### Syntax

```
void FPDOBJArchiveLoaderLoadWideString (
    FPD\_ObjArchiveLoader ar,
    FS\_WideString* pwStr
);
```

### Description

Loads a wide string value from archive.

### Parameter

---

ar	[In] Ref to the input object archive loader.
----	--

---

pwStr	[Out] It receives the wide string value.
-------	--

---

### Return

void

### Head file reference

fpd\_serialTempl.h: 267

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)

## FPDObjArchiveLoaderNew

**Syntax**

```
FPD_ObjArchiveLoader FPDObjArchiveLoaderNew (
    FS\_LPCBYTE pData,
    FS\_DWORD dwSize
);
```

**Description**

Creates a PDF object archive loader.

**Parameter**

---

pData	[In] The input memory block.
-------	------------------------------

---

dwSize	[In] The size of the input memory block.
--------	--

---

**Return**

The PDF object archive loader.

**Head file reference**

fpd\_serialTempl.h: 163

## FPDObjArchiveLoaderRead

**Syntax**

```
FS_BOOL FPDObjArchiveLoaderRead (
    FPD\_ObjArchiveLoader ar,
    ,
);
```

**Description**

De-serializes a memory block.

**Parameter**

---

ar	[In] The input object archive loader.
----	---------------------------------------

---

[In/Out]
----------

---

[In]
------

---

**Return**

non-zero means successful, otherwise failed.

**Head file reference**

fpd\_serialTempl.h: 201

## FPD\_ObjArchiveSaver

**Return from Used by**

### Description

PDF object archive saver class. See [CFPDObjArchiveSaverNew](#) , [CFPDObjArchiveDestroy](#) .

### Returned from

**FPDObjArchiveSaverNew**

### Used by

[FPDObjArchiveDestroy](#)  
[FPDObjArchiveSaverGetBuffer](#)  
[FPDObjArchiveSaverGetLength](#)  
[FPDObjArchiveSaverSaveByte](#)  
[FPDObjArchiveSaverSaveByteString](#)  
[FPDObjArchiveSaverSaveDWORD](#)  
[FPDObjArchiveSaverSaveFloat](#)  
[FPDObjArchiveSaverSaveInteger](#)  
[FPDObjArchiveSaverSaveObject](#)  
[FPDObjArchiveSaverSaveWideString](#)  
[FPDObjArchiveSaverSaveWideStringII](#)  
[FPDObjArchiveSaverWrite](#)

## Functions

### Functions summary

**FPDObjArchiveDestroy**

Destroys the PDF object archive saver.

**FPDObjArchiveSaverGetBuffer**

Gets the constant byte pointer to the saved data.

**FPDObjArchiveSaverGetLength**

Gets the length of saved data.

**FPDObjArchiveSaverNew**

Creates a PDF object archive saver.

**FPDObjArchiveSaverSaveByte**

Overloads operator for serializing a single byte.

**FPDObjArchiveSaverSaveByteString**

Overloads operator for serializing a byte string.

**FPDObjArchiveSaverSaveDWORD**

Overloads operator for serializing a DWORD value.

**FPDOBJArchiveSaverSaveFloat**

Overloads operator for serializing a floating-point.

**FPDOBJArchiveSaverSaveInteger**

Overloads operator for serializing a integer.

**FPDOBJArchiveSaverSaveObject**

Overloads operator for serializing a PDF object.

**FPDOBJArchiveSaverSaveWideString**

Overloads operator for serializing a wide string.

**FPDOBJArchiveSaverSaveWideStringII**

Overloads operator for serializing a wide string.

**FPDOBJArchiveSaverWrite**

Overloads operator for serializing a memory block.

## Functions detail

### FPDOBJArchiveSaverDestroy

**Syntax**

```
void FPDOBJArchiveSaverDestroy (
    FPD_ObjArchiveSaver ar
);
```

**Description**

Destroys the PDF object archive saver.

**Parameter**

---

ar	[In] The input PDF object archive saver.
----	--

---

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 29

### FPDOBJArchiveSaverGetBuffer

**Syntax**

```
FS_LPCBYTE FPDOBJArchiveSaverGetBuffer (
    FPD_ObjArchiveSaver ar
);
```

**Description**

Gets the constant byte pointer to the saved data.

**Parameter**

---

ar	[In] The input PDF object archive saver.
----	--

---

**Return**

The constant byte pointer to the saved data.

**Head file reference**

fpd\_serialTempl.h: 57

**FPDOBJArchiveSaverGetLength****Syntax**

```
FS_INTPTR FPDOBJArchiveSaverGetLength (
    FPD\_ObjArchiveSaver ar
);
```

**Description**

Gets the length of saved data.

**Parameter**

---

ar	[In] The input PDF object archive saver.
----	--

---

**Return**

The length in bytes of the saved data.

**Head file reference**

fpd\_serialTempl.h: 48

**FPDOBJArchiveSaverNew****Syntax**

```
FPD_ObjArchiveSaver FPDOBJArchiveSaverNew (void );
```

**Description**

Creates a PDF object archive saver.

**Return**

The PDF object archive saver.

**Head file reference**

fpd\_serialTempl.h: 21

**FPDOBJArchiveSaverSaveByte****Syntax**

```
void FPDOBJArchiveSaverSaveByte (
```

```
FPD_ObjArchiveSaver ar,  
FS_BYTE i  
);
```

**Description**

Overloads operator for serializing a single byte.

**Parameter**

---

ar	[In] The input PDF object archive saver.
i	[In] The input byte.

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 66

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FPDOBJArchiveSaverSaveByteString****Syntax**

```
void FPDOBJArchiveSaverSaveByteString (  
    FPD_ObjArchiveSaver ar,  
    FS_ByteString str  
) ;
```

**Description**

Overloads operator for serializing a byte string.

**Parameter**

---

ar	[In] The input PDF object archive saver.
str	[In] The input byte string.

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 110

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)**FPDObjArchiveSaverSaveDWORD****Syntax**

```
void FPDObjArchiveSaverSaveDWORD (
```

[FPD\\_ObjArchiveSaver](#) ar,

[FS\\_DWORD](#) i

```
);
```

**Description**

Overloads operator for serializing a DWORD value.

**Parameter**

---

ar	[In] The input PDF object archive saver.
----	--

---

i	[In] The input DWORD value.
---	-----------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 88

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FPDObjArchiveSaverSaveFloat****Syntax**

```
void FPDObjArchiveSaverSaveFloat (
```

[FPD\\_ObjArchiveSaver](#) ar,

[FS\\_FLOAT](#) i

```
);
```

**Description**

Overloads operator for serializing a floating-point.

**Parameter**

---

ar	[In] The input PDF object archive saver.
----	--

---

i	[In] The input floating-point.
---	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 99

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDObjArchiveSaverSaveInteger****Syntax**

```
void FPDObjArchiveSaverSaveInteger (
    FPD\_ObjArchiveSaver ar,
    FS\_INT32 i
);
```

**Description**

Overloads operator for serializing a integer.

**Parameter**

---

ar	[In] The input PDF object archive saver.
----	--

---

i	[In] The input integer.
---	-------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 77

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDObjArchiveSaverSaveObject****Syntax**

```
void FPDObjArchiveSaverSaveObject (
    FPD\_ObjArchiveSaver ar,
    const FPD\_Object pObj
);
```

**Description**

Overloads operator for serializing a PDF object.

**Parameter**

---

ar	[In] Ref to output PDF object archive saver.
----	--

---

pObj	[In] The input PDF object.
------	----------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 38

**FPDObjArchiveSaverSaveWideString****Syntax**

```
void FPDObjArchiveSaverSaveWideString (
    FPD\_ObjArchiveSaver ar,
    FS\_WideString wstr
);
```

**Description**

Overloads operator for serializing a wide string.

**Parameter**


---

ar	[In] The input PDF object archive saver.
----	--

---

wstr	[In] The input wide string.
------	-----------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 121

**Since**[SDK LATEEST VERSION > 1.0](#)**FPDObjArchiveSaverSaveWideStringII****Syntax**

```
void FPDObjArchiveSaverSaveWideStringII (
    FPD\_ObjArchiveSaver ar,
    FS\_LPCWSTR wstr
);
```

**Description**

Overloads operator for serializing a wide string.

**Parameter**

---

ar	[In] The input PDF object archive saver.
----	--

---

wstr	[In] The input wide string.
------	-----------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 132

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)**FPDObjArchiveSaverWrite****Syntax**

```
void FPDObjArchiveSaverWrite (
    FPD\_ObjArchiveSaver ar,
    void* pData,
    FS\_DWORD dwSize
);
```

**Description**

Overloads operator for serializing a memory block.

**Parameter**

---

ar	[In] The input PDF object archive saver.
----	--

---

pData	[In] The pointer to a memory block.
-------	-------------------------------------

---

dwSize	[In] The size in bytes of the memory block.
--------	---

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 143

**Since**[SDK\\_LATEEST\\_VERSION > 1.0](#)

# FPD\_Object

[Return from Used by](#)

## Description

A [FPD Object](#) is a general object in a PDF file, which may be of any object type. This is the abstract class for all PDF syntax objects.  
PDF supports eight basic types of objects:

- - Boolean values [FPD Boolean](#)
- - Integer and real numbers [FPD Number](#)
- - Strings [FPD String](#)
- - Names [FPD Name](#)
- - Arrays [FPD Array](#)
- - Dictionaries [FPD Dictionary](#)
- - Streams [FPD Stream](#)
- - The null object [FPD Null](#)

Addtional type of object: [FPD Reference](#)

. Objects may be labeled so that they can be referred to by other objects. A labeled object is called an indirect object. See [FPDOObjectDestroy](#) , [FPDStringNew](#) , [FPDStringNewW](#) , [FPDNameNew](#) , [FPDBooleanNew](#) , [FPDNumberNewByInt](#) , [FPDNumberNewByFloat](#) , [FPDNumberNewByStr](#) , [FPDNumberNewFromData](#) , [FPDArrayNew](#) , [FPDDictionaryNew](#) , [FPDStreamNew](#) , [FPDNullNew](#) , [FPDRefernceNew](#) , [FPDRefernceNew2](#) .

## Returned from

[FPDFDFDocGetAnnotDict](#)  
[FPDFDFDocGetIndirectObject](#)  
[FPDFDFDocGetRoot](#)  
[FPDFDFDocImportExternalObject](#)  
[FPDFDFDocImportIndirectObject](#)  
[FPDAActionGetDictionary](#)  
[FPDAActionGetAnnot](#)  
[FPDAActionGetDict](#)  
[FPDAActionGetSoundStream](#)  
[FPDAActionFieldsGetField](#)  
[FPDAnnotGetAnnotDict](#)  
[FPDArrayGetArray](#)  
[FPDArrayGetDict](#)  
[FPDArrayGetElement](#)  
[FPDArrayGetValue](#)  
[FPDArrayGetStream](#)  
[FPDArrayNew](#)  
[FPDBookmarkGetDictionary](#)  
[FPDBooleanNew](#)  
[FPDCColorSpaceGetArray](#)  
[FPDDictionaryGetArray](#)  
[FPDDictionaryGetDict](#)  
[FPDDictionaryGetElement](#)  
[FPDDictionaryGetValue](#)  
[FPDDictionaryGetNextElement](#)  
[FPDDictionaryGetStream](#)

[FPDDictionaryNew](#)  
[FPDDocCreateNewPage](#)  
[FPDDocGetIndirectObject](#)  
[FPDDocGetInfo](#)  
[FPDDocGetPage](#)  
[FPDDocGetPageContentModify](#)  
[FPDDocGetRoot](#)  
[FPDDocImportExternalObject](#)  
[FPDDocImportIndirectObject](#)  
[FPDFFileSpecGetFileStream](#)  
[FPDFFontGetFontDict](#)  
[FPDFFontEncodingRealize](#)  
[FPDFFormGetDict](#)  
[FPDFFormGetFormStream](#)  
[FPDFFormGetResourcesDictionary](#)  
[FPDFFormControlGetDownIcon](#)  
[FPDFFormControlGetNormalIcon](#)  
[FPDFFormControlGetRolloverIcon](#)  
[FPDFFormControlGetWidget](#)  
[FPDFFormFieldGetFieldDict](#)  
[FPDIconFitGetDict](#)  
[FPDImageGetDict](#)  
[FPDImageGetOC](#)  
[FPDImageGetStream](#)  
[FPDInlineImagesGetStream](#)  
[FPDInterFormGetFormDict](#)  
[FPDInterFormGetInternalField](#)  
[FPDLWinParamGetDict](#)  
[FPDNameNew](#)  
[FPDNameTreeGetRoot](#)  
[FPDNameTreeLookupNamedDest](#)  
[FPDNameTreeLookupValue](#)  
[FPDNameTreeLookupValueByName](#)  
[FPDNullNew](#)  
[FPDNumberNewByFloat](#)  
[FPDNumberNewByInt](#)  
[FPDNumberNewByStr](#)  
[FPDNumberNewFromData](#)  
[FPDOCGroupGetDictionary](#)  
[FPDOCPropertiesGetConfig](#)  
[FPDPageGetDict](#)  
[FPDPageGetPageAttr](#)  
[FPDPageGetResourcesDictionary](#)  
[FPDParseGetEncryptDict](#)  
[FPDParseGetIDArray](#)  
[FPDParseGetOtherTrailerByIndex](#)  
[FPDParseGetTrailer](#)  
[FPDParseParseIndirectObject](#)  
[FPDParseParseIndirectObjectAt](#)  
[FPDPatternGetPatternObj](#)  
[FPDReferenceNew](#)  
[FPDReferenceNew2](#)  
[FPDShadingPatternGetPatternObj](#)  
[FPDShadingPatternGetShadingObject](#)  
[FPDStreamGetDict](#)  
[FPDStreamNew](#)  
[FPDStreamAccGetDict](#)

[FPDStreamAccGetImageParam](#)  
[FPDStreamAccGetStream](#)  
[FPDStreamFilterGetStream](#)  
[FPDStringNew](#)  
[FPDStringNewW](#)  
[FPDTilingPatternGetPatternObj](#)  
[FPDOObjectClone](#)  
[FPDOObjectCloneRefToDoc](#)  
[FPDOObjectCloneRefToFDFDoc](#)  
[FPDOObjectGetArray](#)  
[FPDOObjectGetContainer](#)  
[FPDOObjectGetDict](#)  
[FPDOObjectGetDirect](#)  
[FPDOObjectParseString](#)

### Used by

[FRAppOpenFileAttachment](#)  
[FRDocSetCustomSecurity](#)  
[FRDocSetDRMSecurity](#)  
[FRPageViewAddAnnot](#)  
[FPDFDFDocAddIndirectObject](#)  
[FPDFDFDocExportAnnotToPDFPage](#)  
[FPDFDFDocGetAnnotPageIndex](#)  
[FPDFDFDocGetNextAssoc](#)  
[FPDFDFDocImportExternalObject](#)  
[FPDFDFDocImportPDFAnnot](#)  
[FPDFDFDocInsertIndirectObject](#)  
[FPDAActionNew](#)  
[FPDActionInsertRendition](#)  
[FPDActionNew](#)  
[FPDActionRemoveRendition](#)  
[FPDActionSetAnnot](#)  
[FPDAnnotNew](#)  
[FPDAnnotListGetAnnotMatrix](#)  
[FPDAnnotListGetAnnotRect](#)  
[FPDArrayAdd](#)  
[FPDArrayAddInteger](#)  
[FPDArrayAddName](#)  
[FPDArrayAddNumber](#)  
[FPDArrayAddReference2ToDoc](#)  
[FPDArrayAddReference2ToFDFDoc](#)  
[FPDArrayAddReferenceToDoc](#)  
[FPDArrayAddReferenceToFDFDoc](#)  
[FPDArrayAddString](#)  
[FPDArrayGetArray](#)  
[FPDArrayGetCount](#)  
[FPDArrayGetDict](#)  
[FPDArrayGetElement](#)  
[FPDArrayGetValue](#)  
[FPDArrayGetFloat](#)  
[FPDArrayGetInteger](#)  
[FPDArrayGetMatrix](#)  
[FPDArrayGetNumber](#)  
[FPDArrayGetRect](#)  
[FPDArrayGetStream](#)

[FPDArrayGetString](#)  
[FPDArrayInsertAt](#)  
[FPDArrayIsIdentical](#)  
[FPDArrayRemoveAt](#)  
[FPDArraySetAt](#)  
[FPDBookmarkNew](#)  
[FPDBooleanIdentical](#)  
[FPDColrSpaceLoad](#)  
[FPDContentMarkAddMark](#)  
[FPDContentMarkLookupMark](#)  
[FPDCreatorSetCustomSecurity](#)  
[FPDCreatorSetDRMSecurity](#)  
[FPDDestGetPDFObject](#)  
[FPDDestNew](#)  
[FPDictionaryAddValue](#)  
[FPDictionaryGetArray](#)  
[FPDictionaryGetBoolean](#)  
[FPDictionaryGetCount](#)  
[FPDictionaryGetDict](#)  
[FPDictionaryGetElement](#)  
[FPDictionaryGetValue](#)  
[FPDictionaryGetFloat](#)  
[FPDictionaryGetInteger](#)  
[FPDictionaryGetInteger2](#)  
[FPDictionaryGetMatrix](#)  
[FPDictionaryGetNextElement](#)  
[FPDictionaryGetNumber](#)  
[FPDictionaryGetRect](#)  
[FPDictionaryGetPosition](#)  
[FPDictionaryGetStream](#)  
[FPDictionaryGetString](#)  
[FPDictionaryGetUnicodeText](#)  
[FPDictionaryIdentical](#)  
[FPDictionaryKeyExist](#)  
[FPDictionaryRemoveAt](#)  
[FPDictionaryReplaceKey](#)  
[FPDictionarySetAt](#)  
[FPDictionarySetAtBoolean](#)  
[FPDictionarySetAtInteger](#)  
[FPDictionarySetAtMatrix](#)  
[FPDictionarySetName](#)  
[FPDictionarySetAtNumber](#)  
[FPDictionarySetAtRect](#)  
[FPDictionarySetAtReference2ToDoc](#)  
[FPDictionarySetAtReference2ToFDFDoc](#)  
[FPDictionarySetAtReferenceToDoc](#)  
[FPDictionarySetAtReferenceToFDFDoc](#)  
[FPDictionarySetAsString](#)  
[FPDDocAddIndirectObject](#)  
[FPDDocConvertIndirectObjects](#)  
[FPDDocGetNextAssoc](#)  
[FPDDocGetPageContentModify](#)  
[FPDDocImportExternalObject](#)  
[FPDDocInsertIndirectObject](#)  
[FPDDocLoadColorSpace](#)  
[FPDDocLoadFont](#)  
[FPDDocLoadFontFile](#)

[FPDDocLoadImageF](#)  
[FPDDocLoadPattern](#)  
[FPDFFileSpecNewFromObj](#)  
[FPDFFontNew](#)  
[FPDFFontEncodingLoadEncoding](#)  
[FPDFFormNew](#)  
[FPDFFormRealizeResource](#)  
[FPDFFormSetResourcesDictionary](#)  
[FPDGeneralStateSetSoftMask](#)  
[FPDIconFitNew](#)  
[FPDImageLoadDIBitmapProgressive](#)  
[FPDImageLoadImageF](#)  
[FPDInlineImagesSetStream](#)  
[FPDInterFormGetControlByDict](#)  
[FPDInterFormGetFieldByDict](#)  
[FPDLLinkNew](#)  
[FPDLWinParamNew](#)  
[FPDMediaPlayerNewFromDict](#)  
[FPDMeshStreamNew](#)  
[FPDNameGetString](#)  
[FPDNameIdentical](#)  
[FPDNameTreeNew](#)  
[FPDNameTreeSetValue](#)  
[FPDNumberGetFloat](#)  
[FPDNumberGetInteger](#)  
[FPDNumberGetNumber](#)  
[FPDNumberGetString](#)  
[FPDNumberIdentical](#)  
[FPDNumberIsInteger](#)  
[FPDNumberSetNumber](#)  
[FPDNumberSetString](#)  
[FPDOBJArchiveLoaderLoadObject](#)  
[FPDOCContextSetOCGState](#)  
[FPDOCGroupNew](#)  
[FPDOCGroupSetNew](#)  
[FPDOCPropertiesIsOCGInPage](#)  
[FPDPageGetPageText](#)  
[FPDPageGetPageText\\_Unicode](#)  
[FPDPageLoad](#)  
[FPDPageRealizeResource](#)  
[FPDPageSetResourcesDictionary](#)  
[FPDPageRenderCacheGetCachedBitmap](#)  
[FPDPageRenderCacheResetBitmap](#)  
[FPDParseLoadAttachmentStream](#)  
[FPDParseReloadFileStream](#)  
[FPDReferenceGetRefObjNum](#)  
[FPDReferenceIdentical](#)  
[FPDReferenceSetRefToDoc](#)  
[FPDReferenceSetRefToFDFDoc](#)  
[FPDRenderContextNew2](#)  
[FPDRenditionNewFromDict](#)  
[FPDShadingPatternNew](#)  
[FPDStreamGetDict](#)  
[FPDStreamGetRawSize](#)  
[FPDStreamGetStreamFilter](#)  
[FPDStreamIdentical](#)  
[FPDStreamIsMemoryBased](#)

[FPDStreamReadRawData](#)  
[FPDStreamSetData](#)  
[FPDStreamAccLoadAllData](#)  
[FPDStringGetString](#)  
[FPDStringIdentical](#)  
[FPDStringIsHex](#)  
[FPDStringSetHex](#)  
[FPDTilingPatternNew](#)  
[FPDTilingPatternNewII](#)  
[FPDTType3FontSetPageResources](#)  
[FPDOObjectClone](#)  
[FPDOObjectCloneRefToDoc](#)  
[FPDOObjectCloneRefToFDFDoc](#)  
[FPDOObjectDestroy](#)  
[FPDOObjectGetArray](#)  
[FPDOObjectGetContainer](#)  
[FPDOObjectGetDict](#)  
[FPDOObjectGetDirect](#)  
[FPDOObjectGetDirectType](#)  
[FPDOObjectGetInteger](#)  
[FPDOObjectGetNumber](#)  
[FPDOObjectGetObjNum](#)  
[FPDOObjectGetString](#)  
[FPDOObjectGetType](#)  
[FPDOObjectGetUnicodeText](#)  
[FPDOObjectIsIdentical](#)  
[FPDOObjectIsModified](#)  
[FPDOObjectSetContainer](#)  
[FPDOObjectSetModified](#)  
[FPDOObjectSetString](#)  
[FPDOObjectSetUnicodeText](#)

## Definitions

### Definitions summary

#### [FPD\\_OBJ\\_ARRAY](#)

Array object.

#### [FPD\\_OBJ\\_BOOLEAN](#)

Boolean object.

#### [FPD\\_OBJ\\_DICTIONARY](#)

Dictionary object.

#### [FPD\\_OBJ\\_INVALID](#)

invalid object.

#### [FPD\\_OBJ\\_NAME](#)

Name object.

#### [FPD\\_OBJ\\_NULL](#)

Null object.

#### [FPD\\_OBJ\\_NUMBER](#)

Number object.

#### [FPD\\_OBJ\\_REFERENCE](#)

Reference object.

#### [FPD\\_OBJ\\_STREAM](#)

Stream object.

**FPD\_OBJ\_STRING**

String object.

**Definitions detail****FPD\_OBJ\_ARRAY****Syntax**

```
#define FPD_OBJ_ARRAY 5
```

**Description**

Array object.

**Group**

[FPDObjTypes](#)

**Head file reference**

fpd\_objsExpT.h: 56

**FPD\_OBJ\_BOOLEAN****Syntax**

```
#define FPD_OBJ_BOOLEAN 1
```

**Description**

Boolean object.

**Group**

[FPDObjTypes](#)

**Head file reference**

fpd\_objsExpT.h: 48

**FPD\_OBJ\_DICTIONARY****Syntax**

```
#define FPD_OBJ_DICTIONARY 6
```

**Description**

Dictionary object.

**Group**

[FPDObjTypes](#)

**Head file reference**

fpd\_objsExpT.h: 58

## FPD\_OBJ\_INVALID

**Syntax**

```
#define FPD_OBJ_INVALID 0
```

**Description**

invalid object.

**Group**

[FPDObjTypes](#)

**Head file reference**

fpd\_objsExpT.h: 46

## FPD\_OBJ\_NAME

**Syntax**

```
#define FPD_OBJ_NAME 4
```

**Description**

Name object.

**Group**

[FPDObjTypes](#)

**Head file reference**

fpd\_objsExpT.h: 54

## FPD\_OBJ\_NULL

**Syntax**

```
#define FPD_OBJ_NULL 8
```

**Description**

Null object.

**Group**

[FPDObjTypes](#)

**Head file reference**

fpd\_objsExpT.h: 62

## FPD\_OBJ\_NUMBER

**Syntax**

```
#define FPD_OBJ_NUMBER 2
```

**Description**

Number object.

**Group**

[FPDObjTypes](#)

**Head file reference**

fpd\_objsExpT.h: 50

## FPD\_OBJ\_REFERENCE

**Syntax**

```
#define FPD_OBJ_REFERENCE 9
```

**Description**

Reference object.

**Group**

[FPDObjTypes](#)

**Head file reference**

fpd\_objsExpT.h: 64

## FPD\_OBJ\_STREAM

**Syntax**

```
#define FPD_OBJ_STREAM 7
```

**Description**

Stream object.

**Group**

[FPDObjTypes](#)

**Head file reference**

fpd\_objsExpT.h: 60

## FPD\_OBJ\_STRING

**Syntax**

```
#define FPD_OBJ_STRING 3
```

**Description**

String object.

**Group**

[FPDObjTypes](#)**Head file reference**

fpd\_objsExpT.h: 52

## Functions

### Functions summary

[FPDOObjectClone](#)

Gets a complete clone. The bDirect param specifies whether a totally direct copy is requested (without any reference inside, so the copy can be copied to other document).

[FPDOObjectCloneRefToDoc](#)

Gets a clone for direct object, or a reference for document.

[FPDOObjectCloneRefToFDFDoc](#)

Gets a clone for direct object, or a reference for memory document.

[FPDOObjectDestroy](#)

Destroys the object. This function has no effect to indirect objects.

[FPDOObjectGetArray](#)

Gets array value of the object. Applicable to arrays only. [NULL](#) is returned for non-array objects.

[FPDOObjectGetContainer](#)

Gets the container of this object.

[FPDOObjectGetDict](#)

Gets dictionary value of the object. Applicable to dictionaries or stream only. [NULL](#) is returned for other types of objects.

[FPDOObjectGetDirect](#)

Gets direct object (the object data itself) of an object. For direct object, return itself; for reference object, return the referred object.

[FPDOObjectGetDirectType](#)

Gets type of direct object without loading it.

[FPDOObjectGetInteger](#)

Gets integer number value of the object. Applicable to number and boolean objects. If object type not supported, 0 is returned.

[FPDOObjectGetNumber](#)

Gets float number value of the object. Applicable to number objects only. If object type is not number, 0.0f is returned. FIX: when FIX format is used, the returned value will be in FIX24.8 format.

[FPDOObjectGetObjNum](#)

Gets indirect number of the object. 0 for direct object.

[FPDOObjectGetString](#)

Gets string value of the object. Applicable to string, name, and number objects. If object type not supported, empty string is returned.

[FPDOObjectGetType](#)

Gets type of the object.

[FPDOObjectGetUnicodeText](#)

Gets Unicode text value of the object. Applicable to string and stream objects. If object type not supported, empty string is returned. We assume the original text are encoding in PDF text encoding scheme. The returned text is encoded in UTF-16LE encoding. A character mapper can be used to convert the original text (if not already encoded in Unicode). If no character mapper used, PDFDocEncoding mapping is used.

**FPDObjectIsIdentical**

Compares with another object.

**FPDObjectIsModified**

Tests whether the object has been "modified".

**FPDObjectParseString**

Parses an object from a memory buffer.

**FPDObjectSetContainer**

Sets the container of this object.

**FPDObjectSetModified**

Changes the object's "modified" flag.

**FPDObjectSetString**

Sets string value into the object. Applicable to boolean, number, string and name objects.

For non-supported object types, this function does nothing.

**FPDObjectSetUnicodeText**

Sets text encoded in Unicode (UTF-16LE format). Applicable to string and stream objects.  
"len" is number of characters, not bytes. -1 for null terminated string.

## Functions detail

### FPDObjectClone

**Syntax**

```
FPD_Object FPDObjectClone (
    FPD Object srcObj,
    FS\_BOOL bDirect
);
```

**Description**

Gets a complete clone. The bDirect param specifies whether a totally direct copy is requested (without any reference inside, so the copy can be copied to other document).

**Parameter**

srcObj	[In] The input PDF syntax objects.
--------	------------------------------------

bDirect	[In] Whether a totally direct copy is requested.
---------	--

**Return**

A complete clone object.

**Head file reference**

fpd\_objsTempl.h: 49

### FPDObjectCloneRefToDoc

**Syntax**

```
FPD_Object FPDObjectCloneRefToDoc (
    FPD Object srcObj,
    FPD Document others
```

);

**Description**

Gets a clone for direct object, or a reference for document.

**Parameter**

---

srcObj	[In] The input PDF syntax objects.
--------	------------------------------------

---

others	[In] The document.
--------	--------------------

---

**Return**

A clone direct object or a reference for document.

**Head file reference**

fpd\_objsTempl.h: 60

**FPDObjectCloneRefToFDFDoc****Syntax**

```
FPD_Object FPDObjectCloneRefToFDFDoc (
    FPD Object srcObj,
    FDF Document others
);
```

**Description**

Gets a clone for direct object, or a reference for memory document.

**Parameter**

---

srcObj	[In] The input PDF syntax objects.
--------	------------------------------------

---

others	[In] The memory document.
--------	---------------------------

---

**Return**

A clone direct object or a reference for memory document.

**Head file reference**

fpd\_objsTempl.h: 70

**FPDObjectDestroy****Syntax**

```
void FPDObjectDestroy (
    FPD Object obj
```

);

**Description**

Destroys the object. This function has no effect to indirect objects.

**Parameter**

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 90

**FPDObjectGetArray****Syntax**

```
FPD_Object FPDObjectGetArray (
    FPD Object obj
);
```

**Description**

Gets array value of the object. Applicable to arrays only. [NULL](#) is returned for non-array objects.

**Parameter**

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

**Return**

The array value of the object.

**Head file reference**

fpd\_objsTempl.h: 156

**FPDObjectGetContainer****Syntax**

```
FPD_Object FPDObjectGetContainer (
    FPD Object obj
);
```

**Description**

Gets the container of this object.

**Parameter**

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

**Return**

The container of this object.

**Head file reference**

fpd\_objsTempl.h: 217

**FPDObjectGetDict****Syntax**

```
FPD_Object FPDObjectGetDict (
    FPD\_Object obj
);
```

**Description**

Gets dictionary value of the object. Applicable to dictionaries or stream only. [NULL](#) is returned for other types of objects.

**Parameter**

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

**Return**

The dictionary value of the object.

**Head file reference**

fpd\_objsTempl.h: 146

**FPDObjectGetDirect****Syntax**

```
FPD_Object FPDObjectGetDirect (
    FPD\_Object obj
);
```

**Description**

Gets direct object (the object data itself) of an object. For direct object, return itself; for reference object, return the referred object.

**Parameter**

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

**Return**

The direct object (the object data itself) of an object.

**Head file reference**

fpd\_objsTempl.h: 80

**FPDObjectGetDirectType****Syntax**

```
FS_INT32 FPDObjectGetDirectType (
    FPD Object obj
);
```

**Description**

Gets type of direct object without loading it.

**Parameter**

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

**Return**

The type of direct object without loading it.

**Head file reference**

fpd\_objsTempl.h: 189

**FPDObjectGetInteger****Syntax**

```
FS_INT32 FPDObjectGetInteger (
    FPD Object obj
);
```

**Description**

Gets integer number value of the object. Applicable to number and boolean objects. If object type not supported, 0 is returned.

**Parameter**

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

**Return**

The integer number value of the object.

**Head file reference**

fpd\_objsTempl.h: 136

## FPDObjectGetNumber

### Syntax

```
FS_FLOAT FPDObjectGetNumber (
    FPD Object obj
);
```

### Description

Gets float number value of the object. Applicable to number objects only. If object type is not number, 0.0f is returned. FIX: when FIX format is used, the returned value will be in FIX24.8 format.

### Parameter

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

### Return

The float number value of the object.

### Head file reference

fpd\_objsTempl.h: 125

## FPDObjectGetobjNum

### Syntax

```
FS_DWORD FPDObjectGetobjNum (
    FPD Object obj
);
```

### Description

Gets indirect number of the object. 0 for direct object.

### Parameter

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

### Return

The indirect number of the object.

### Head file reference

fpd\_objsTempl.h: 30

## FPDObjectGetString

### Syntax

```
void FPDObjectGetString (
    FPD Object obj,
    FS ByteString* outString
```



);

**Description**

Gets string value of the object. Applicable to string, name, and number objects. If object type not supported, empty string is returned.

**Parameter**

obj	[In] The input PDF syntax objects.
outString	[Out] It receives the string value of the object.

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 99

**FPDOBJECTGETTYPE****Syntax**

```
FS_INT32 FPDOBJECTGETTYPE (
    FPD_Object obj
);
```

**Description**

Gets type of the object.

**Parameter**

obj	[In] The input PDF syntax objects.
-----	------------------------------------

**Return**

One of the FPD\_OBJ\_xxxx constants.

**Head file reference**

fpd\_objsTempl.h: 21

**FPDOBJECTGETUNICODETEXT****Syntax**

```
void FPDOBJECTGETUNICODETEXT (
    FPD_Object obj,
    FS_WideString* outUnicodeText
);
```

**Description**

Gets Unicode text value of the object. Applicable to string and stream objects. If object type not supported, empty string is returned. We assume the original text are encoding in PDF text encoding scheme. The returned text is encoded in UTF-16LE encoding. A character mapper can be used to convert the original text (if not already encoded in Unicode). If no character mapper used, PDFDocEncoding mapping is used.

**Parameter**

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

outUnicodeText	[Out] An Unicode text value.
----------------	------------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 110

**FPDObjectIsIdentical****Syntax**

```
FS_BOOL FPDObjectIsIdentical (
    FPD Object obj,
    FPD Object otherObj
);
```

**Description**

Compares with another object.

**Parameter**

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

otherObj	[In] The input object.
----------	------------------------

---

**Return**

Non-zero means identical, otherwise not identical.

**Head file reference**

fpd\_objsTempl.h: 39

**FPDObjectIsModified****Syntax**

```
FS_BOOL FPDOBJECTIsModified (
    FPD Object obj
);
```

**Description**

Tests whether the object has been "modified".

**Parameter**

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

**Return**

[TRUE](#) for the object having been "modified".

**Head file reference**

fpd\_objsTempl.h: 198

**FPDOBJECTParseString****Syntax**

```
FPD_OBJECT FPDOBJECTParseString (
    FS LPCSTR str
);
```

**Description**

Parses an object from a memory buffer.

**Parameter**

---

str	[In] A string containing the object
-----	-------------------------------------

---

**Return**

A parsed object, or [NULL](#) if error.

**Head file reference**

fpd\_objsTempl.h: 236

**Note:** Parsing indirect reference inside the string will bring undefined result.

**FPDOBJECTSetContainer****Syntax**

```
void FPDOBJECTSetContainer (
    FPD Object obj,
    FPD Object container
);
```

**Description**

Sets the container of this object.

**Parameter**

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

---

container	[In] The container of this object.
-----------	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 226

**FPDObjectSetModified****Syntax**

```
void FPDObjectSetModified (
    FPD Object obj,
    FS\_BOOL bModified
);
```

**Description**

Changes the object's "modified" flag.

**Parameter**

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

---

bModified	[In] The "modified" flag. TRUE means "modified".
-----------	--

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 207

**FPDObjectSetString****Syntax**

```
void FPDObjectSetString (
    FPD Object obj,
    FS LPCSTR str
);
```

**Description**

Sets string value into the object. Applicable to boolean, number, string and name objects. For non-supported object types, this function does nothing.

**Parameter**

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

str	[In] The input string value.
-----	------------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 166

**FPDObjectSetUnicodeText****Syntax**

```
void FPDObjectSetUnicodeText (
    FPD Object obj,
    FS LPCWSTR str,
    FS INT32 len
);
```

**Description**

Sets text encoded in Unicode (UTF-16LE format). Applicable to string and stream objects. "len" is number of characters, not bytes. -1 for null terminated string.

**Parameter**

---

obj	[In] The input PDF syntax objects.
-----	------------------------------------

---

str	[In] Pointer to UTF-16LE format characters.
-----	---

---

len	[In] Number of the input characters.
-----	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 177

**FPD\_OCContext**

[\*\*Return from Used by\*\*](#)**Description**

A FPD\_OCContext is an object that keeps track the on/off states of all of the OCGs a document. There can be more than one FPD\_OCContext object, representing different combinations of OCG states. The FPD\_Document contains an internal FPD\_OCContext that is used for on-screen drawing and as the default state used for any other drawing or content enumeration.

Clients can change the states of OCGs within any FPD\_OCContext. Clients can build (and save in the PDF file) FPD\_OCContext objects with their own combination of OCG states, and issue drawing or enumeration commands using their own FPD\_OCContext instead of the document's internal FPD\_OCContext. See [FPDOCContextNew](#), [FPDOCContextDestroy](#), [FPDOCContextGetDocument](#), [FPDOCContextGetUsageType](#), [FPDOCContextCheckOCGVisible](#), [FPDOCContextResetOCContext](#).

**Returned from**

[FRDocViewGetOCContext](#)  
[FPDRenderOptionsCreateOCContextHandler](#)  
[FPDOCContextNew](#)

**Used by**

[FPDRenderOptionsCreateOCContextHandler](#)  
[FPDRenderOptionsDeleteOCContextHandler](#)  
[FPDRenderOptionsSetOCCHandler](#)  
[FPDOCContextCheckOCGVisible](#)  
[FPDOCContextDestroy](#)  
[FPDOCContextGetDocument](#)  
[FPDOCContextGetUsageType](#)  
[FPDOCContextResetOCContext](#)  
[FPDOCContextSetOCGState](#)

**Enumerations****Enumerations summary**[\*\*FPD OCC UsageType\*\*](#)

The usage type of optional content. See [FPDOCContextNew](#).

[\*\*FPD OCGState\*\*](#)

State, for action type SetOCGState. See [FPDACTIONCOUNTOCGSTATES](#), [FPDACTIONGETOCGSTATES](#), [FPDACTIONINSERTOCGSTATES](#), [FPDACTIONREPLACEOCGSTATES](#), [FPDACTIONREMOVEOCGSTATES](#).

**Enumerations detail****FPD\_OCC\_UsageType****Syntax**

```
enum FPD_OCC_UsageType{
    View,
    Design,
    Print,
    Export}
```

```
};
```

**Description**

The usage type of optional content. See [FPDOCContextNew](#) .

**Head file reference**

fpd\_docExpT.h: 393

**View**

Used for a viewer .

**Design**

Used to represent a document designer's structural organization of artwork.

**Print**

Used for printing.

**Export**

Used for exporting.

## FPD\_OCGState

**Syntax**

```
enum FPD_OCGState{  
    ON,  
    OFF,  
    Toggle  
};
```

**Description**

State, for action type SetOCGState. See [FPDACTIONCOUNTOCGSTATES](#) , [FPDACTIONGETOCGSTATES](#) , [FPDACTIONINSERTOCGSTATES](#) , [FPDACTIONREPLACEOCGSTATES](#) , [FPDACTIONREMOVEOCGSTATES](#) .

**Head file reference**

fpd\_docExpT.h: 411

**ON**

Sets the state of subsequent groups to ON.

**OFF**

Sets the state of subsequent groups to OFF.

**Toggle**

Reverses the state of subsequent groups.

## Functions

### Functions summary

**FPDOCContextCheckOCGVisible**

Checks whether the optional content group is visible or not.

**FPDOCContextDestroy**

Destroys optional content context.

**FPDOCContextGetDocument**

Gets the PDF document in the optional content context.

**FPDOCContextGetUsageType**

Gets the usage type of optional content.

**FPDOCContextNew**

Creates optional content context from a PDF document.

**FPDOCContextResetOCCContext**

Resets the optional content context.

**FPDOCContextSetOCGState**

Resets the optional content context.

## Functions detail

### FPDOCContextCheckOCGVisible

**Syntax**

```
FS_BOOL FPDOCContextCheckOCGVisible (
    FPD_OCContext occ,
    const FPD_Object OCGDict
);
```

**Description**

Checks whether the optional content group is visible or not.

**Parameter**

occ	[In] The input optional content context.
-----	--

OCGDict	[In] The optional content group dictionary.
---------	---

**Return**

TRUE for being visible.

**Head file reference**

fpd\_docTempl.h: 1074

### FPDOCContextDestroy

**Syntax**

```
void FPDOCContextDestroy (
    FPD_OCContext occ
);
```

**Description**

Destroys optional content context.

**Parameter**

---

occ	[In] Optional content context.
-----	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1047

**FPDOCContextGetDocument****Syntax**

```
FPD_Document FPDOCContextGetDocument (
    FPD\_OCContext occ
);
```

**Description**

Gets the PDF document in the optional content context.

**Parameter**

---

occ	[In] The input optional content context.
-----	--

---

**Return**

The PDF document in the optional content context.

**Head file reference**

fpd\_docTempl.h: 1056

**FPDOCContextGetUsageType****Syntax**

```
FS_INT32 FPDOCContextGetUsageType (
    FPD\_OCContext occ
);
```

**Description**

Gets the usage type of optional content.

**Parameter**

---

occ	[In] The input optional content context.
-----	--

---

**Return**

The usage type of optional content.

**Head file reference**

fpd\_docTempl.h: 1065

**FPDOCContextNew****Syntax**

```
FPD_OCContext FPDOCContextNew (
    FPD Document doc,
    FPD OCC UsageType UsageType
);
```

**Description**

Creates optional content context from a PDF document.

**Parameter**

---

doc	[In] The input PDF document.
-----	------------------------------

---

UsageType	[In] The intended usage type of optional content.
-----------	---

---

**Return**

An optional content context.

**Head file reference**

fpd\_docTempl.h: 1037

**FPDOCContextResetOCContext****Syntax**

```
void FPDOCContextResetOCContext (
    FPD OCContext occ
);
```

**Description**

Resets the optional content context.

**Parameter**

---

occ	[In] The input optional content context.
-----	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1084

**FPDOCContextSetOCGState****Syntax**

```
void FPDOCContextSetOCGState (
    FPD\_OCContext occ,
    FPD\_Object ocgDict,
    FS\_BOOL bState,
    FS\_BOOL bNotify
);
```

**Description**

Resets the optional content context.

**Parameter**

occ	[In] The input optional content context.
ocgDict	[In] The input OCG dictionary.
bState	[In] The new state of the OCG.
bNotify	[In] Whether to notify or not.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1093

**FPD\_OCContextHandler****Return from Used by****Description**

PDF optional content context interface. Used for holding a [FPD\\_OCContextCallBack](#).

**Returned from****[FPDRenderOptionsCreateOCContextHandler](#)****Used by**

[\*\*FPDRenderOptionsDeleteOCCContextHandler\*\*](#)  
[\*\*FPDRenderOptionsSetOCCHandler\*\*](#)

## Callbacks

### Callbacks summary

[\*\*CheckOCGVisible\*\*](#)

Check whether a option content group is visible.

[\*\*CheckObjectVisible\*\*](#)

Check whether an object is visible in this context

### Callbacks detail

#### CheckOCGVisible

**Syntax**

```
typedef FS_BOOL (*CheckOCGVisible)(  
    FPD\_Object ogc  
);
```

**Description**

Check whether a option content group is visible.

**Parameter**

---

ogc	[In] The optional content group dictionary.
-----	---

---

**Return**

Non-zero means visible, otherwise invisible.

**Head file reference**

fpd\_renderExpT.h: 227

**Group**

[\*\*FPD\\_OCContextCallBack\*\*](#)

#### CheckObjectVisible

**Syntax**

```
typedef FS_BOOL (*CheckObjectVisible)(  
    FPD\_PageObject obj  
);
```

**Description**

Check whether an object is visible in this context

**Parameter**

---

obj	[In] Page object.
-----	-------------------

---

**Return**

Non-zero means visible, otherwise invisible.

**Head file reference**

fpd\_renderExpT.h: 237

**Group**

[FPD\\_OCContextCallBack](#)

## FPD\_OCGroup

**[Return from Used by](#)**

### Description

A FPD\_OCGroup represents a named object whose state can be toggled in a user interface to affect changes in visibility of content. The FPD\_OCGroup object represents an optional-content group. This corresponds to a PDF OCG dictionary representing a collection of graphic objects that can be made visible or invisible. Any graphic content of the PDF can be made optional, including page contents, XObjects, and rotations. The specific content objects in the group have an OC entry in the PDF. See [FPDOCGroupNew](#) , [FPDOCGroupDestroy](#) , [FPDOCGroupSetIsSubGroupSet](#) .

### Returned from

[FPDOCGroupSetNew](#)  
[FPDOCGroupNew](#)

### Used by

[FPDOCGroupSetCountElements](#)  
[FPDOCGroupSetDestroy](#)  
[FPDOCGroupSetFindGroup](#)  
[FPDOCGroupSetGetGroup](#)  
[FPDOCGroupSetGetSubGroupSet](#)  
[FPDOCGroupSetGetSubGroupSetName](#)  
[FPDOCGroupSetIsSubGroupSet](#)  
[FPDOCPropertiesGetOCGroupOrder](#)  
[FPDOCGroupDestroy](#)  
[FPDOCGroupGetCreatorInfo](#)  
[FPDOCGroupGetDictionary](#)  
[FPDOCGroupGetExportState](#)  
[FPDOCGroupGetLanguageInfo](#)  
[FPDOCGroupGetName](#)  
[FPDOCGroupGetPageElementType](#)  
[FPDOCGroupGetPrintInfo](#)  
[FPDOCGroup GetUser Type](#)  
[FPDOCGroupGetViewState](#)  
[FPDOCGroupGetZoomRange](#)  
[FPDOCGroupHasIntent](#)

**FPDOCGroupSetName**

## Callbacks

### Callbacks summary

**FPD\_OCGStateChangedNotify**

A callback for Optional Content Notification [FPD\\_OCNotify](#) object. Triggered when OCGstate has been changed.

### Callbacks detail

**FPD\_OCGStateChangedNotify****Syntax**

```
typedef void (*FPD_OCGStateChangedNotify)(  
    FPD\_Document doc,  
    FPD\_OCGroup ogc,  
    FS\_BOOL bVisible  
)
```

**Description**

A callback for Optional Content Notification [FPD\\_OCNotify](#) object. Triggered when OCGstate has been changed.

**Parameter**

doc	[In] The PDF document.
-----	------------------------

ogc	[In] The optional content group object.
-----	---

bVisible	[In] Whether the OCG is visible.
----------	----------------------------------

**Return**

void

**Head file reference**

fpd\_docExpT.h: 468

## Functions

### Functions summary

**FPDOCGroupDestroy**

Destroys optional content group;

**FPDOCGroupGetCreatorInfo**

CreateInfo entry of optional content usage dictionary.

**FPDOCGroupGetDictionary**

Get the OCG dictionary.

#### **FPDOCGroupGetExportState**

Gets the export state of the OCG. Exports entry of optional content usage dictionary.

#### **FPDOCGroupGetLanguageInfo**

Language entry of optional content usage dictionary.

#### **FPDOCGroupGetName**

Gets the name of the OCG.

#### **FPDOCGroupGetPageElementType**

Gets PageElement entry of optional content usage dictionary.

#### **FPDOCGroupGetPrintInfo**

Gets print entry of optional content usage dictionary.

#### **FPDOCGroup GetUserType**

User entry of optional content usage dictionary.

#### **FPDOCGroupGetViewState**

Gets the view state of the OCG. View entry of optional content usage dictionary.

#### **FPDOCGroupGetZoomRange**

Gets zoom entry of optional content usage dictionary.

#### **FPDOCGroupHasIntent**

Does the OCG have the specified intent?

#### **FPDOCGroupNew**

Creates optional content group from a PDF dictionary.

#### **FPDOCGroupSetName**

Sets the name of the OCG.

## Functions detail

### FPDOCGroupDestroy

#### **Syntax**

```
void FPDOCGroupDestroy (
    FPD\_OCGroup ogc
);
```

#### **Description**

Destroys optional content group;

#### **Parameter**

ogc	[In] Optional content group.
-----	------------------------------

#### **Return**

void

#### **Head file reference**

fpd\_docTempl.h: 1122

### FPDOCGroupGetCreatorInfo

#### **Syntax**

```
FS_BOOL FPDOCGroupGetCreatorInfo (
    FPD\_OCGROUP ogc,
    FS\_WideString* outWstrCreator,
    FS\_ByteString* outStrType
);
```

**Description**

CreatorInfo entry of optional content usage dictionary.

**Parameter**

ocg	[In] Optional content group.
outWstrCreator	[Out] It receives that specifies the application that created the group.
outStrType	[Out] It receives that specifies the type of content controlled by the group.

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 1161

**FPDOCGroupGetDictionary****Syntax**

```
FPD_Object FPDOCGroupGetDictionary (
    FPD\_OCGROUP ogc
);
```

**Description**

Get the OCG dictionary.

**Parameter**

ocg	[In] Optional content group.
-----	------------------------------

**Return**

The OCG dictionary.

**Head file reference**

fpd\_docTempl.h: 1244

## FPDOCGroupGetExportState

### Syntax

```
FS_BOOL FPDOCGroupGetExportState (
    FPD\_OCGroup ogc
);
```

### Description

Gets the export state of the OCG. Exports entry of optional content usage dictionary.

### Parameter

---

ogc	[In] Optional content group.
-----	------------------------------

---

### Return

The export state of the OCG.

### Head file reference

fpd\_docTempl.h: 1183

## FPDOCGroupGetLanguageInfo

### Syntax

```
FS_BOOL FPDOCGroupGetLanguageInfo (
    FPD\_OCGroup ogc,
    FS\_ByteString* outInfo,
    FS\_BOOL* outPreferred
);
```

### Description

Language entry of optional content usage dictionary.

### Parameter

---

ogc	[In] Optional content group.
-----	------------------------------

---

---

outInfo	[Out] It receives that specifies a language and possibly a locale.
---------	--

---

---

outPreferred	[Out] It receives whether the language is a preferred language.
--------------	---

---

### Return

Non-zero means success, otherwise failure.

### Head file reference

fpd\_docTempl.h: 1172

## FPDOCGroupGetName

### Syntax

```
FS_BOOL FPDOCGroupGetName (
    FPD\_OCGroup ogc,
    FS\_WideString* outName
);
```

### Description

Gets the name of the OCG.

### Parameter

---

ocg	[In] Optional content group.
-----	------------------------------

---

outName	[Out] It will receive the name of the OCG.
---------	--

---

### Return

Non-zero means success, otherwise failure.

### Head file reference

fpd\_docTempl.h: 1131

## FPDOCGroupGetPageElementType

### Syntax

```
FS_BOOL FPDOCGroupGetPageElementType (
    FPD\_OCGroup ogc,
    FS\_ByteString* outName
);
```

### Description

Gets PageElement entry of optional content usage dictionary.

### Parameter

---

ocg	[In] Optional content group.
-----	------------------------------

---

outName	[Out] It receives the page element type.
---------	--

---

### Return

Non-zero means success, otherwise failure.

### Head file reference

fpd\_docTempl.h: 1234

**FPDOCGroupGetPrintInfo****Syntax**

```
FS_BOOL FPDOCGroupGetPrintInfo (
    FPD_OCGroup ogc,
    FS_ByteString* outType,
    FS_BOOL* outState
);
```

**Description**

Gets print entry of optional content usage dictionary.

**Parameter**

ocg	[In] Optional content group.
outType	[Out] It receives that specifies the kind of content controlled by the group.
outState	[Out] It receives the printing state of OCG.

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 1203

**FPDOCGroup GetUserType****Syntax**

```
FS_BOOL FPDOCGroup GetUserType (
    FPD_OCGroup ogc,
    FS_ByteString* outType,
    FS_WideStringArray* outUserArr
);
```

**Description**

User entry of optional content usage dictionary.

**Parameter**

ocg	[In] Optional content group.
outType	[Out] It receives the user type of whom this OCG is primarily intended.

outUserArr	[Out] It receives a string that represents the name(s) of the individual, position or organization.
------------	---

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 1223

---

**FPDOCGroupGetViewState****Syntax**

```
FS_BOOL FPDOCGroupGetViewState (
    FPD\_OCGROUP ogc
);
```

**Description**

Gets the view state of the OCG. View entry of optional content usage dictionary.

**Parameter**

---

ocg	[In] Optional content group.
-----	------------------------------

---

**Return**

The view state of the OCG.

**Head file reference**

fpd\_docTempl.h: 1214

---

**FPDOCGroupGetZoomRange****Syntax**

```
FS_BOOL FPDOCGroupGetZoomRange (
    FPD\_OCGROUP ogc,
    FS\_FLOAT* outMin,
    FS\_FLOAT* outMax
);
```

**Description**

Gets zoom entry of optional content usage dictionary.

**Parameter**

---

ocg	[In] Optional content group.
-----	------------------------------

---

---

outMin	[Out] It receives the minimum recommended magnification factor.
--------	---

---

outMax	[Out] It receives the maximum recommended magnification factor
--------	--

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 1192

**FPDOCGroupHasIntent****Syntax**

```
FS_BOOL FPDOCGroupHasIntent (
    FPD\_OCGroup ogc,
    FS\_LPCSTR szIntent
);
```

**Description**

Does the OCG have the specified intent?

**Parameter**


---

ocg	[In] Optional content group.
-----	------------------------------

---

szIntent	[In] The input intent.
----------	------------------------

---

**Return**

Whether the OCG has the specified intent or not.

**Head file reference**

fpd\_docTempl.h: 1151

**FPDOCGroupNew****Syntax**

```
FPD_OCGroup FPDOCGroupNew (
    FPD\_Object dict
);
```

**Description**

Creates optional content group from a PDF dictionary.

**Parameter**

---

dict	[In] The input PDF dictionary.
------	--------------------------------

---

**Return**

Optional content group from a PDF dictionary.

**Head file reference**

fpd\_docTempl.h: 1113

**FPDOCGroupSetName****Syntax**

```
void FPDOCGroupSetName (
    FPD\_OCGROUP ogc,
    FS\_LPCWSTR wszName
);
```

**Description**

Sets the name of the OCG.

**Parameter**


---

ogc	[In] Optional content group.
-----	------------------------------

---

wszName	[In] The new OCG name.
---------	------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1141

**FPD\_OCGroupSet****[Return from Used by](#)****Description**

A set of mutually exclusive OCGs. See [FPDOCGroupSetNew](#) , [FPDOCGroupSetDestroy](#) , [FPDOCGroupSetIsSubGroupSet](#) , [FPDOCGroupSetGetGroup](#) , [FPDOCGroupSetFindGroup](#) .

**Returned from****[FPDOCGroupSetNew](#)****Used by**

---

[\*\*FPDOCPropertiesGetOCGroupOrder\*\*](#)  
[\*\*FPDOCGroupSetCountElements\*\*](#)  
[\*\*FPDOCGroupSetDestroy\*\*](#)  
[\*\*FPDOCGroupSetFindGroup\*\*](#)  
[\*\*FPDOCGroupSetGetGroup\*\*](#)  
[\*\*FPDOCGroupSetGetSubGroupSet\*\*](#)  
[\*\*FPDOCGroupSetGetSubGroupSetName\*\*](#)  
[\*\*FPDOCGroupSetIsSubGroupSet\*\*](#)

## Functions

### Functions summary

[\*\*FPDOCGroupSetCountElements\*\*](#)

Gets the count of elements in the OCG set.

[\*\*FPDOCGroupSetDestroy\*\*](#)

Destroys optional content group set.

[\*\*FPDOCGroupSetFindGroup\*\*](#)

Finds a OCG in the array.

[\*\*FPDOCGroupSetGetGroup\*\*](#)

Gets a OCG from specified position.

[\*\*FPDOCGroupSetGetSubGroupSet\*\*](#)

Gets a OCG set from specified position.

[\*\*FPDOCGroupSetGetSubGroupSetName\*\*](#)

Gets the group set name.

[\*\*FPDOCGroupSetIsSubGroupSet\*\*](#)

Checks whether the specified element is a subgroup or not.

[\*\*FPDOCGroupSetNew\*\*](#)

Creates optional content group set from a PDF object.

### Functions detail

#### FPDOCGroupSetCountElements

##### Syntax

```
FS_INT32 FPDOCGroupSetCountElements (
    FPD\_OCGGroupSet ocgs
);
```

##### Description

Gets the count of elements in the OCG set.

##### Parameter

---

ocgs	[In] Optional content group set.
------	----------------------------------

---

##### Return

The count of elements in the OCG set.

##### Head file reference

fpd\_docTempl.h: 1279

## FPDOCGroupSetDestroy

### Syntax

```
void FPDOCGroupSetDestroy (
    FPD\_OCGroupSet ocgs
);
```

### Description

Destroys optional content group set.

### Parameter

---

ocgs	[In] Optional content group set.
------	----------------------------------

---

### Return

void

### Head file reference

fpd\_docTempl.h: 1270

## FPDOCGroupSetFindGroup

### Syntax

```
FS_INT32 FPDOCGroupSetFindGroup (
    FPD\_OCGroupSet ocgs,
    const FPD\_Object groupDict
);
```

### Description

Finds a OCG in the array.

### Parameter

---

ocgs	[In] Optional content group set.
------	----------------------------------

---

groupDict	[In] The input OCG dictionary.
-----------	--------------------------------

---

### Return

The zero-based index in the array.

### Head file reference

fpd\_docTempl.h: 1320

## FPDOCGroupSetGetGroup

**Syntax**

```
void FPDOCGroupSetGetGroup (
    FPD\_OCGroupSet ocgs,
    FS\_INT32 index,
    FPD\_OCGroup* pOCGroup
);
```

**Description**

Gets a OCG from specified position.

**Parameter**


---

ocgs	[In] Optional content group set.
------	----------------------------------

---

index	[In] The input zero-based element in the array.
-------	---

---

pOCGroup	[Out] It receives the optional content group.
----------	---

---

**Return**

void

**Head file reference**

[fpd\\_docTempl.h](#): 1298

**FPDOCGroupSetGetSubGroupSet****Syntax**

```
void FPDOCGroupSetGetSubGroupSet (
    FPD\_OCGroupSet ocgs,
    FS\_INT32 index,
    FPD\_OCGroupSet* pOCGroupSet
);
```

**Description**

Gets a OCG set from specified position.

**Parameter**


---

ocgs	[In] Optional content group set.
------	----------------------------------

---

index	[In] The input zero-based element in the array.
-------	---

---

pOCGroupSet	[Out] It receives the optional content group set.
-------------	---

---

**Return**

---

```
void
```

**Head file reference**

fpd\_docTempl.h: 1309

**FPDOCGroupSetGetSubGroupSetName****Syntax**

```
FS_BOOL FPDOCGroupSetGetSubGroupSetName (
    FPD_OCGroupSet ocgs,
    FS_WideString* outName
);
```

**Description**

Gets the group set name.

**Parameter**


---

ocgs	[In] Optional content group set.
------	----------------------------------

---

outName	[Out] It receives the name of the OCG set.
---------	--

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 1330

**FPDOCGroupSetIsSubGroupSet****Syntax**

```
FS_BOOL FPDOCGroupSetIsSubGroupSet (
    FPD_OCGroupSet ocgs,
    FS_INT32 index
);
```

**Description**

Checks whether the specified element is a subgroup or not.

**Parameter**


---

ocgs	[In] Optional content group set.
------	----------------------------------

---

index	[In] The input zero-based element in the array.
-------	---

---

**Return**

Non-zero means a subgroup, otherwise not.

**Head file reference**

fpd\_docTempl.h: 1288

## FPDOCGroupSetNew

**Syntax**

```
FPD_OCGroupSet FPDOCGroupSetNew (
    FPD\_Object obj
);
```

**Description**

Creates optional content group set from a PDF object.

**Parameter**

---

obj	[In] A PDF object.
-----	--------------------

---

**Return**

An optional content group set.

**Head file reference**

fpd\_docTempl.h: 1261

## FPD\_OCNotify

**Return from Used by****Description**

Optional Content Notification Interface. See [FPDOCNotifyFPD\\_OCNotifyNew](#) , [FPDOCNotifyFPD\\_OCNotifyDestroy](#) , [FPDOCPropertiesAddOCNotify](#) , [FPDOCPropertiesRemoveOCNotify](#) .

**Returned from****FPDOCNotifyFPD\_OCNotifyNew****Used by**

[FPDOCPropertiesAddOCNotify](#)  
[FPDOCPropertiesRemoveOCNotify](#)  
[FPDOCNotifyFPD\\_OCNotifyDestroy](#)

**Functions**

## Functions summary

### [FPDOCNotifyFPD\\_OCNotifyDestroy](#)

Destroys the OC notify interface.

### [FPDOCNotifyFPD\\_OCNotifyNew](#)

Creates a new OC notify interface.

## Functions detail

### FPDOCNotifyFPD\_OCNotifyDestroy

#### Syntax

```
void FPDOCNotifyFPD_OCNotifyDestroy (
    FPD\_OCNotify notify
);
```

#### Description

Destroys the OC notify interface.

#### Parameter

---

notify	[In] The OC notify interface.
--------	-------------------------------

---

#### Return

void

#### Head file reference

fpd\_docTempl.h: 1357

### FPDOCNotifyFPD\_OCNotifyNew

#### Syntax

```
FPD_OCNotify FPDOCNotifyFPD_OCNotifyNew (
    FPD\_OCGStateChangedNotify proc
);
```

#### Description

Creates a new OC notify interface.

#### Parameter

---

proc	[In] A callback for Optional Content Notification FPD_OCNotify object.
------	---

---

#### Return

The OC notify interface.

#### Head file reference

fpd\_docTempl.h: 1348

## FPD\_OCProperties

[Return from](#) [Used by](#)

### Description

The optional content properties dictionary for a document. See [FPDOCPropertiesNew](#) , [FPDOCPropertiesDestroy](#) , [FPDOCPropertiesCountConfigs](#) , [FPDOCPropertiesGetConfig](#) , [FPDOCPropertiesAddOCNotify](#) , [FPDOCPropertiesRemoveOCNotify](#) .

### Returned from

[FPDOCPropertiesNew](#)

### Used by

[FPDOCPropertiesAddOCNotify](#)  
[FPDOCPropertiesCountConfigs](#)  
[FPDOCPropertiesDestroy](#)  
[FPDOCPropertiesGetConfig](#)  
[FPDOCPropertiesGetDocument](#)  
[FPDOCPropertiesGetOCGroupOrder](#)  
[FPDOCPropertiesGetOCGroups](#)  
[FPDOCPropertiesIsOCGInPage](#)  
[FPDOCPropertiesIsOCGroup](#)  
[FPDOCPropertiesRemoveOCNotify](#)  
[FPDOCPropertiesRetrieveOCGPages](#)

### Functions

#### Functions summary

[FPDOCPropertiesAddOCNotify](#)

Adds an user-supplied optional content notify interface to the OCP.

[FPDOCPropertiesCountConfigs](#)

Gets the count of configuration dictionaries in the OCP.

[FPDOCPropertiesDestroy](#)

Destroys optional content properties.

[FPDOCPropertiesGetConfig](#)

Gets a configuration dictionary in the OCP.

[FPDOCPropertiesGetDocument](#)

Gets the PDF document.

[FPDOCPropertiesGetOCGroupOrder](#)

Orders entry in optional content configuration dictionary. All document level OCG objects can be stored in an ordered tree object, this will be showed in UI.

[FPDOCPropertiesGetOCGroups](#)

Retrieves all OCG objects for a document or a page. If iPageIndex equals to -1, all document level OCG objects are returned; or all page level OCG objects are returned.

**FPDOCPropertiesIsOCGInPage**

Determines whether a OCG object is in a page or not.

**FPDOCPropertiesIsOCGroup**

Determines whether a dictionary object is a valid OCG object.

**FPDOCPropertiesNew**

Creates optional content properties from a PDF document.

**FPDOCPropertiesRemoveOCNotify**

Removes an user-supplied optional content notify interface from OCP.

**FPDOCPropertiesRetrieveOCGPages**

Retrieves all pages objects in which the specified pOCGDict is referenced. One OCG can be shared by several pages. pages is an array of page dictionary objects. The returned value is the count of the elements in pages array.

## Functions detail

### FPDOCPropertiesAddOCNotify

#### Syntax

```
void FPDOCPropertiesAddOCNotify (
    FPD\_OCProperties ocprops,
    FPD\_OCNNotify ogNotifyCallback
);
```

#### Description

Adds an user-supplied optional content notify interface to the OCP.

#### Parameter

ocprops	[In] Optional content properties.
---------	-----------------------------------

ogNotifyCallback	[In] The input user supplied notify interface to add.
------------------	---

#### Return

void

#### Head file reference

fpd\_docTempl.h: 1478

### FPDOCPropertiesCountConfigs

#### Syntax

```
FS_INT32 FPDOCPropertiesCountConfigs (
    FPD\_OCProperties ocprops
);
```

#### Description

Gets the count of configuration dictionaries in the OCP.

**Parameter**

---

ocprops	[In] Optional content properties.
---------	-----------------------------------

---

**Return**

The count of configuration dictionaries in the OCP.

**Head file reference**

fpd\_docTempl.h: 1459

**FPDOCPropertiesDestroy****Syntax**

```
void FPDOCPropertiesDestroy (
    FPD\_OCProperties ocprops
);
```

**Description**

Destroys optional content properties.

**Parameter**

---

ocprops	[In] Optional content properties.
---------	-----------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1383

**FPDOCPropertiesGetConfig****Syntax**

```
FPD_Object FPDOCPropertiesGetConfig (
    FPD\_OCProperties ocprops,
    FS\_INT32 index
);
```

**Description**

Gets a configuration dictionary in the OCP.

**Parameter**

---

ocprops	[In] Optional content properties.
---------	-----------------------------------

---

---

index	[In] The input zero-based configuration dictionary index.
-------	---

---

**Return**

A configuration dictionary.

**Head file reference**

fpd\_docTempl.h: 1468

**FPDOCPropertiesGetDocument****Syntax**

```
FPD_Document FPDOCPropertiesGetDocument (
    FPD\_OCProperties ocprops
);
```

**Description**

Gets the PDF document.

**Parameter**

---

ocprops	[In] Optional content properties.
---------	-----------------------------------

---

**Return**

The PDF document.

**Head file reference**

fpd\_docTempl.h: 1392

**FPDOCPropertiesGetOCGroupOrder****Syntax**

```
void FPDOCPropertiesGetOCGroupOrder (
    FPD\_OCProperties ocprops,
    FPD\_OCGroupSet* pOCGroupSet
);
```

**Description**

Orders entry in optional content configuration dictionary. All document level OCG objects can be stored in an ordered tree object, this will be showed in UI.

**Parameter**

---

ocprops	[In] Optional content properties.
---------	-----------------------------------

---

---

pOCGroupSet	[Out] It receives the OCG set.
-------------	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1448

**FPDOCPropertiesGetOCGroups****Syntax**

```
FS_INT32 FPDOCPropertiesGetOCGroups (
    FPD\_OCProperties oprops,
    FS\_PtrArray* arrOCGs,
    FS\_INT32 page_index
);
```

**Description**

Retrieves all OCG objects for a document or a page. If iPageIndex equals to -1, all document level OCG objects are returned; or all page level OCG objects are returned.

**Parameter**

---

oprops	[In] Optional content properties.
--------	-----------------------------------

---

arrOCGs	[Out] It receives all OCG objects in specified page.
---------	--

---

page_index	[In] The input zero-based page index.
------------	---------------------------------------

---

**Return**

The number of OCG retrieved.

**Head file reference**

fpd\_docTempl.h: 1401

**FPDOCPropertiesIsOCGInPage****Syntax**

```
FS_BOOL FPDOCPropertiesIsOCGInPage (
    FPD\_OCProperties oprops,
    const FPD\_Object page_dict,
    FPD\_Object ogc_dict
);
```

**Description**

Determines whether a OCG object is in a page or not.

**Parameter**

---

ocprops	[In] Optional content properties.
page_dict	[In] The input page dictionary.
ocg_dict	[In] The input OCG dictionary.

---

**Return**

Non-zero means in the page, otherwise not.

**Head file reference**

fpd\_docTempl.h: 1437

**FPDOCPropertiesIsOCGroup****Syntax**

```
FS_BOOL FPDOCPropertiesIsOCGroup (
    FPD\_OCProperties ocprops,
    const FPD\_Object dict
);
```

**Description**

Determines whether a dictionary object is a valid OCG object.

**Parameter**


---

ocprops	[In] Optional content properties.
dict	[In] The input PDF dictionary.

---

**Return**

Non-zero means valid, otherwise invalid.

**Head file reference**

fpd\_docTempl.h: 1414

**FPDOCPropertiesNew****Syntax**

```
FPD_OCProperties FPDOCPropertiesNew (
    FPD\_Document doc
);
```

**Description**

Creates optional content properties from a PDF document.

**Parameter**


---

doc	[In] The input PDF document.
-----	------------------------------

---

**Return**

Optional content properties.

**Head file reference**

fpd\_docTempl.h: 1374

**FPDOCPropertiesRemoveOCNotify****Syntax**

```
void FPDOCPropertiesRemoveOCNotify (
    FPD\_OCProperties ocprops,
    FPD\_OCNotify ogNotifyCallback
);
```

**Description**

Removes an user-supplied optional content notify interface from OCP.

**Parameter**


---

ocprops	[In] Optional content properties.
---------	-----------------------------------

---

ogNotifyCallback	[In] The input user supplied notify interface to remove.
------------------	--

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 1488

**FPDOCPropertiesRetrieveOCGPages****Syntax**

```
FS_INT32 FPDOCPropertiesRetrieveOCGPages (
    FPD\_OCProperties ocprops,
    const FPD\_Object dict,
    FS\_PtrArray* arrPages
);
```

**Description**

Retrieves all pages objects in which the specified pOCGDict is referenced. One OCG can be shared by several pages. pages is an array of page dictionary objects. The returned value is the count of the elements in pages array.

**Parameter**


---

ocprops	[In] Optional content properties.
dict	[In] The input OCG dictionary.
arrPages	[Out] It receives all page dictionaries in which the specified OCG is referenced.

---

**Return**

The number of pages referenced.

**Head file reference**

fpd\_docTempl.h: 1424

## FPD\_Page

### [Return from Used by](#)

### Description

A [FPD\\_Page](#) is a page in a document, corresponding to the PDF Page object (see Page Objects in Section 3.6.2, Page Tree, in the PDF Reference). Among other associated objects, a page contains:

- A series of objects representing the objects drawn on the page.
- A list of resources used in drawing the page.
- Annotations.
- An optional thumbnail image of the page.
- Beads used in any articles that occur on the page.

See [FPDPageNew](#) , [FPDPageDestroy](#) .

### Returned from

[FRPageViewGetPDPage](#)  
[FRScrollBarThumbnailViewGetPDPage](#)  
[FRThumbnailViewGetPDPage](#)  
[FPDClipPathGetText](#)  
[FPDFFormGetNextObject](#)  
[FPDFFormGetObjectAt](#)  
[FPDFFormGetObjectByIndex](#)  
[FPDFFormGetPrevObject](#)  
[FPDFFormObjectNew](#)  
[FPDIImageObjectNew](#)  
[FPDInlineImagesNew](#)  
[FPDPageArchiveLoaderNew](#)  
[FPDPageArchiveSaverNew](#)  
[FPDPageObjectClone](#)

[FPDPageObjectNew](#)  
[FPDPageRenderCacheNew](#)  
[FPDPathObjectNew](#)  
[FPDRenderContextGetPageCache](#)  
[FPDShadingObjectGetPage](#)  
[FPDShadingObjectNew](#)  
[FPDTextObjectNew](#)  
[FPDPageGetNextObject](#)  
[FPDPageGetObjectAt](#)  
[FPDPageGetObjectByIndex](#)  
[FPDPageGetPrevObject](#)  
[FPDPageGetRenderCache](#)  
[FPDPageNew](#)

Used by

[FPDFDFDocExportAnnotToPDFPage](#)  
[FPDAnnotListNew](#)  
[FPDClipPathAppendTexts](#)  
[FPDFFormGetObjectIndex](#)  
[FPDFFormInsertObject](#)  
[FPDFFormReplaceObject](#)  
[FPDFFormControlDrawControl](#)  
[FPDFFormObjectCalcBoundingBox](#)  
[FPDFFormObjectDestroy](#)  
[FPDFFormObjectGetForm](#)  
[FPDFFormObjectGetTransformMatrix](#)  
[FPDFFormObjectSetForm](#)  
[FPDFFormObjectSetTransformMatrix](#)  
[FPDFFormObjectTransform](#)  
[FPDImageResetCache](#)  
[FPDImageObjectCalcBoundingBox](#)  
[FPDImageObjectDestroy](#)  
[FPDImageObjectGetImage](#)  
[FPDImageObjectGetTransformMatrix](#)  
[FPDImageObjectSetImage](#)  
[FPDImageObjectSetTransformMatrix](#)  
[FPDImageObjectTransform](#)  
[FPDInlineImagesAddMatrix](#)  
[FPDInlineImagesCountMatrix](#)  
[FPDInlineImagesDestroy](#)  
[FPDInlineImagesGetMatrix](#)  
[FPDInlineImagesGetStream](#)  
[FPDInlineImagesSetStream](#)  
[FPDInterFormCountPageControls](#)  
[FPDInterFormFixPageFields](#)  
[FPDInterFormGetControlAtPoint](#)  
[FPDInterFormGetPageControl](#)  
[FPDLinkCountLinks](#)  
[FPDLinkGetLink](#)  
[FPDLinkGetLinkAtPoint](#)  
[FPDPageArchiveLoaderDestroy](#)  
[FPDPageArchiveLoaderLoadClipPath](#)  
[FPDPageArchiveLoaderLoadColorState](#)  
[FPDPageArchiveLoaderLoadGeneralState](#)  
[FPDPageArchiveLoaderLoadGraphState](#)

[FPDPageArchiveLoaderLoadObject](#)  
[FPDPageArchiveLoaderLoadTextState](#)  
[FPDPageArchiveLoaderNew](#)  
[FPDPageArchiveSaverDestroy](#)  
[FPDPageArchiveSaverNew](#)  
[FPDPageArchiveSaverSaveClipPath](#)  
[FPDPageArchiveSaverSaveColorState](#)  
[FPDPageArchiveSaverSaveGeneralState](#)  
[FPDPageArchiveSaverSaveGraphState](#)  
[FPDPageArchiveSaverSavePageObject](#)  
[FPDPageArchiveSaverSaveTextState](#)  
[FPDPageObjectAppendClipPath](#)  
[FPDPageObjectClone](#)  
[FPDPageObjectCopy](#)  
[FPDPageObjectCopyClipPath](#)  
[FPDPageObjectCopyStates](#)  
[FPDPageObjectDefaultStates](#)  
[FPDPageObjectDestroy](#)  
[FPDPageObjectGetBBox](#)  
[FPDPageObjectGetClipPath](#)  
[FPDPageObjectGetColorState](#)  
[FPDPageObjectGetContentMark](#)  
[FPDPageObjectGetContentMark2](#)  
[FPDPageObjectGetGeneralState](#)  
[FPDPageObjectGetGraphState](#)  
[FPDPageObjectGetOriginalBBox](#)  
[FPDPageObjectGetTextState](#)  
[FPDPageObjectGetType](#)  
[FPDPageObjectHasClipPath](#)  
[FPDPageObjectRemoveClipPath](#)  
[FPDPageObjectSetColorState](#)  
[FPDPageObjectSetGeneralState](#)  
[FPDPageObjectSetGraphState](#)  
[FPDPageObjectSetTextState](#)  
[FPDPageObjectTransformClipPath](#)  
[FPDPageRenderCacheClearAll](#)  
[FPDPageRenderCacheClearImageData](#)  
[FPDPageRenderCacheDestroy](#)  
[FPDPageRenderCacheEstimateSize](#)  
[FPDPageRenderCacheGetCachedBitmap](#)  
[FPDPageRenderCacheNew](#)  
[FPDPageRenderCacheResetBitmap](#)  
[FPDPathObjectCalcBoundingBox](#)  
[FPDPathObjectDestroy](#)  
[FPDPathObjectGetFillMode](#)  
[FPDPathObjectGetPath](#)  
[FPDPathObjectGetTransformMatrix](#)  
[FPDPathObjectIsStrokeMode](#)  
[FPDPathObjectSetFillMode](#)  
[FPDPathObjectSetGraphState](#)  
[FPDPathObjectSetStrokeMode](#)  
[FPDPathObjectSetTransformMatrix](#)  
[FPDPathObjectTransform](#)  
[FPDProgressiveSearchFindFrom](#)  
[FPDRenderContextAppendPage](#)  
[FPDRenderContextDrawPage](#)  
[FPDRenderContextNew](#)

[FPDRenderContextNew2](#)  
[FPDRenderDeviceDrawType3Text](#)  
[FPDShadingObjectCalcBoundingBox](#)  
[FPDShadingObjectDestroy](#)  
[FPDShadingObjectGetPage](#)  
[FPDShadingObjectGetShadingPattern](#)  
[FPDShadingObjectGetTransformMatrix](#)  
[FPDShadingObjectSetPage](#)  
[FPDShadingObjectSetShadingPattern](#)  
[FPDShadingObjectSetTransformMatrix](#)  
[FPDShadingObjectTransform](#)  
[FPDTextObjectCalcCharPos](#)  
[FPDTextObjectCountChars](#)  
[FPDTextObjectCountItems](#)  
[FPDTextObjectDestroy](#)  
[FPDTextObjectGetCharInfo](#)  
[FPDTextObjectGetData](#)  
[FPDTextObjectGetFont](#)  
[FPDTextObjectGetFontSize](#)  
[FPDTextObjectGetItemInfo](#)  
[FPDTextObjectGetPosX](#)  
[FPDTextObjectGetPosY](#)  
[FPDTextObjectGetTextMatrix](#)  
[FPDTextObjectRecalcPositionData](#)  
[FPDTextObjectSetData](#)  
[FPDTextObjectSetEmpty](#)  
[FPDTextObjectSetPosition](#)  
[FPDTextObjectSetText](#)  
[FPDTextObjectSetText2](#)  
[FPDTextObjectSetText3](#)  
[FPDTextObjectSetTextState](#)  
[FPDTextObjectTransform](#)  
[FPDTextPageNew](#)  
[FPDPageBackgroundAlphaNeeded](#)  
[FPDPageCalcBoundingBox](#)  
[FPDPageClearRenderCache](#)  
[FPDPageContinueParse](#)  
[FPDPageCountObjects](#)  
[FPDPageDebugOutput](#)  
[FPDPageDestroy](#)  
[FPDPageEstimateParseProgress](#)  
[FPDPageFindCSName](#)  
[FPDPageFindFontName](#)  
[FPDPageGenerateContent](#)  
[FPDPageGetDict](#)  
[FPDPageGetDisplayMatrix](#)  
[FPDPageGetDocument](#)  
[FPDPageGetFirstObjectPosition](#)  
[FPDPageGetLastObjectPosition](#)  
[FPDPageGetNextObject](#)  
[FPDPageGetObjectAt](#)  
[FPDPageGetObjectByIndex](#)  
[FPDPageGetObjectIndex](#)  
[FPDPageGetPageAttr](#)  
[FPDPageGetPageBBox](#)  
[FPDPageGetPageHeight](#)  
[FPDPageGetPageMatrix](#)

[\*\*FPDPageGetPageWidth\*\*](#)  
[\*\*FPDPageGetParseState\*\*](#)  
[\*\*FPDPageGetPrevObject\*\*](#)  
[\*\*FPDPageGetRenderCache\*\*](#)  
[\*\*FPDPageGetResourcesDictionary\*\*](#)  
[\*\*FPDPageInsertObject\*\*](#)  
[\*\*FPDPageIsParsed\*\*](#)  
[\*\*FPDPageLoad\*\*](#)  
[\*\*FPDPageMoveObject\*\*](#)  
[\*\*FPDPageNewContentGenerator\*\*](#)  
[\*\*FPDPageParseContent\*\*](#)  
[\*\*FPDPageRealizeResource\*\*](#)  
[\*\*FPDPageRemoveObject\*\*](#)  
[\*\*FPDPageReplaceObject\*\*](#)  
[\*\*FPDPageSetResourcesDictionary\*\*](#)  
[\*\*FPDPageStartParse\*\*](#)  
[\*\*FPDPageTransform\*\*](#)

## Functions

### Functions summary

[\*\*FPDPageBackgroundAlphaNeeded\*\*](#)

Checks whether this object list needs background alpha channel to render.

[\*\*FPDPageCalcBoundingBox\*\*](#)

Calculates the bounding box of all page objects in the collection.

[\*\*FPDPageClearRenderCache\*\*](#)

Clears render cache.

[\*\*FPDPageContinueGenerateContent\*\*](#)

Continues to generate the PDF content progressively.

[\*\*FPDPageContinueParse\*\*](#)

Continue parsing.

[\*\*FPDPageCountObjects\*\*](#)

Gets the count of page objects in the collection.

[\*\*FPDPageDebugOutput\*\*](#)

Outputs debug information. For debug only: list all page objects.

[\*\*FPDPageDestroy\*\*](#)

Destroys the PDF page.

[\*\*FPDPageDestroyContentGenerator\*\*](#)

Destroys the PDF page content generator.

[\*\*FPDPageEstimateParseProgress\*\*](#)

Estimates the percentage of parse progress.

[\*\*FPDPageFindCSName\*\*](#)

Finds a resource name of specified color space.

[\*\*FPDPageFindFontName\*\*](#)

Finds a resource name of specified font.

[\*\*FPDPageGenerateContent\*\*](#)

Replaces the page content stream.

[\*\*FPDPageGetDict\*\*](#)

Gets the page dictionary.

[\*\*FPDPageGetDisplayMatrix\*\*](#)

Builds a matrix from PDF user space to targeted device space, according to metrics info (top-left position and width-height size) provided in device space.

**[FPDPageGetDocument](#)**

Gets the PDF document.

**[FPDPageGetFirstObjectPosition](#)**

Gets the position of the first page object.

**[FPDPageGetLastObjectPosition](#)**

Gets the position of the last page object

**[FPDPageGetNextObject](#)**

Gets the current object and moves to next position.

**[FPDPageGetObjectAt](#)**

Gets an object at specified position.

**[FPDPageGetObjectByIndex](#)**

Gets an object by a zero-based page object index.

**[FPDPageGetObjectIndex](#)**

Gets the zero-based page object index in the object array.

**[FPDPageGetPageAttr](#)**

Gets an inheritable attribute value.

**[FPDPageGetPageBBox](#)**

Gets the page bounding box in user space.

**[FPDPageGetPageHeight](#)**

Gets the page height in user space.

**[FPDPageGetPageMatrix](#)**

Gets the page matrix.

**[FPDPageGetPageText](#)**

Extracts pure text from a page, in appearance order.

**[FPDPageGetPageText\\_Unicode](#)**

Extracts Unicode pure text from a page, in appearance order.

**[FPDPageGetPageWidth](#)**

Gets the page width in user space.

**[FPDPageGetParseState](#)**

Gets the current parsing state.

**[FPDPageGetPrevObject](#)**

Gets the current object and moves to previous position

**[FPDPageGetRenderCache](#)**

Gets render cache. for RENDER module.

**[FPDPageGetResourcesDictionary](#)**

Gets the PDF page resources dictionary.

**[FPDPageInsertObject](#)**

Inserts a page object. To insert before all objects, use NULL for *posInsertAfter* .

**[FPDPageIsParsed](#)**

Checks whether the content has been parsed into page objects.

**[FPDPageLoad](#)**

Constructs a page. For saving memory, the page caching feature can be disabled, then images and masks used in page rendering won't be cached. Of course this will affect the speed.

**[FPDPageMoveObject](#)**

Moves a page object from a position to another position.

**[FPDPageNew](#)**

Creates a new empty PDF page.

**[FPDPageNewContentGenerator](#)**

Creates the PDF page content generator.

**FPDPageParseContent**

Parses all contents.

**FPDPageRealizeResource**

Adds a resource to current object list. Returns the resource name.

**FPDPageRemoveObject**

Removes a page object.

**FPDPageReplaceObject**

Replaces a page object with a new page object.

**FPDPageSetResourcesDictionary**

Sets the PDF page resources dictionary.

**FPDPageStartGenerateContent**

Starts to generate the PDF content progressively.

**FPDPageStartParse**

Start parsing the page. If pausing is enabled, application should check current parsing state after this function returns. If parsing not finished, ContinueParse() should be called.

**FPDPageTransform**

Transforms all objects.

## Functions detail

### FPDPageBackgroundAlphaNeeded

**Syntax**

```
FS_BOOL FPDPageBackgroundAlphaNeeded (
    FPD Page page
);
```

**Description**

Checks whether this object list needs background alpha channel to render.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

TRUE for this object list needing background alpha channel to render.

**Head file reference**

fpd\_pageTempl.h: 219

**Note:** If it's TRUE, then the application should better use ARGB device to render it, otherwise the objects may need more time to render. Please call this function after the page has been parsed.

### FPDPageCalcBoundingBox

**Syntax**

```
FS_FloatRect FPDPageCalcBoundingBox (
```



```
FPD_Page page  
);
```

**Description**

Calculates the bounding box of all page objects in the collection.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

The bounding box of all page objects in the collection.

**Head file reference**

fpd\_pageTempl.h: 230

**FPDPageClearRenderCache****Syntax**

```
void FPDPageClearRenderCache (  
    FPD_Page page  
,
```

**Description**

Clears render cache.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 400

**FPDPageContinueGenerateContent****Syntax**

```
FPD_ProgressiveStatus FPDPageContinueGenerateContent (  
    FPD_ContentGenerator generator,  
    FS_PauseHandler pause  
,
```

**Description**

Continues to generate the PDF content progressively.

**Parameter**

---

generator	[In] The input PDF content generator.
pause	[In] The input pause handler. Creates the pause handler by <a href="#">FSPauseHandlerCreate</a> . Sets it NULL as default.

---

**Return**

The status of generating PDF page content progressively.

**Head file reference**

fpd\_pageTempl.h: 472

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FPDPageContinueParse****Syntax**

```
void FPDPageContinueParse (
    FPD\_Page page,
    FS\_PauseHandler pauseHandler
);
```

**Description**

Continue parsing.

**Parameter**

---

page	[In] The input PDF page.
pauseHandler	[In] The user-supplied pause handler.

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 39

**FPDPageCountObjects****Syntax**

```
FS_DWORD FPDPageCountObjects (
    FPD\_Page page
);
```

**Description**

Gets the count of page objects in the collection.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

The count of page objects in the collection.

**Head file reference**

fpd\_pageTempl.h: 127

**FPDPageDebugOutput****Syntax**

```
void FPDPageDebugOutput (
    FPD Page page,
    FS\_LPSTR szFileName
);
```

**Description**

Outputs debug information. For debug only: list all page objects.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

---

szFileName	[In] The input file name.
------------	---------------------------

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 199

**FPDPageDestroy****Syntax**

```
void FPDPageDestroy (
    FPD Page page
);
```

**Description**

Destroys the PDF page.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 30

**FPDPageDestroyContentGenerator****Syntax**

```
void FPDPageDestroyContentGenerator (
    FPD\_ContentGenerator generator
);
```

**Description**

Destroys the PDF page content generator.

**Parameter**

---

generator	[In] The input PDF content generator.
-----------	---------------------------------------

---

**Return**

void.

**Head file reference**

fpd\_pageTempl.h: 450

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FPDPageEstimateParseProgress****Syntax**

```
FS_INT32 FPDPageEstimateParseProgress (
    FPD\_Page page
);
```

**Description**

Estimates the percentage of parse progress.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

The percentage of parse progress.

**Head file reference**

fpd\_pageTempl.h: 70

**FPDPageFindCSName****Syntax**

```
void FPDPageFindCSName (
    FPD Page page,
    FPD ColorSpace cs,
    FS ByteString* outResName
);
```

**Description**

Finds a resource name of specified color space.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

---

cs	[In] The input color space.
----	-----------------------------

---

---

outResName	[Out] It receive the resource name of the color space.
------------	--

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 274

**FPDPageFindFontName****Syntax**

```
void FPDPageFindFontName (
    FPD Page page,
    FPD Font font,
    FS ByteString* outResName
);
```

**Description**

Finds a resource name of specified font.

**Parameter**

page	[In] The input PDF page.
font	[In] The input font.
outResName	[Out] It receive the resource name of the font.

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 285

**FPDPageGenerateContent****Syntax**

```
void FPDPageGenerateContent (
    FPD\_Page page
);
```

**Description**

Replaces the page content stream.

**Parameter**

page	[In] The input PDF page.
------	--------------------------

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 409

**FPDPageGetDict****Syntax**

```
FPD_Object FPDPageGetDict (
    FPD\_Page page
);
```

**Description**

Gets the page dictionary.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

The page dictionary.

**Head file reference**

fpd\_pageTempl.h: 239

**FPDPageGetDisplayMatrix****Syntax**

```
FS_AffineMatrix FPDPageGetDisplayMatrix (
    FPD Page page,
    FS INT32 xPos,
    FS INT32 yPos,
    FS INT32 xSize,
    FS INT32 ySize,
    FS INT32 iRotate
);
```

**Description**

Builds a matrix from PDF user space to targeted device space, according to metrics info (top-left position and width-height size) provided in device space.

**Parameter**


---

page	[In] The input PDF page.
------	--------------------------

---

xPos	[In] The x-coordinate of the top-left position in the device space.
------	---

---

yPos	[In] The y-coordinate of the top-left position in the device space.
------	---

---

xSize	[In] The x-direction size in the device space.
-------	--

---

ySize	[In] The y-direction size in the device space.
-------	--

---

iRotate	[In] The rotation degrees.
---------	----------------------------

---

**Return**

The transformation matrix from PDF user space to targeted device space.

**Head file reference**

fpd\_pageTempl.h: 330

## FPDPageGetDocument

### Syntax

```
FPD_Document FPDPageGetDocument (
    FPD Page page
);
```

### Description

Gets the PDF document.

### Parameter

---

page	[In] The input PDF page.
------	--------------------------

---

### Return

The PDF document.

### Head file reference

fpd\_pageTempl.h: 248

## FPDPageGetFirstObjectPosition

### Syntax

```
FS_POSITION FPDPageGetFirstObjectPosition (
    FPD Page page
);
```

### Description

Gets the position of the first page object.

### Parameter

---

page	[In] The input PDF page.
------	--------------------------

---

### Return

The position of the first page object.

### Head file reference

fpd\_pageTempl.h: 79

## FPDPageGetLastObjectPosition

### Syntax

```
FS_POSITION FPDPageGetLastObjectPosition (
    FPD Page page
);
```

);

**Description**

Gets the position of the last page object

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

The position of the last page object

**Head file reference**

fpd\_pageTempl.h: 88

**FPDPageGetNextObject****Syntax**

```
FPD_PageObject FPDPageGetNextObject (
    FPD Page page,
    FS\_POSITION* inOutPos
);
```

**Description**

Gets the current object and moves to next position.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

inOutPos	[In/Out] The input current position, and receives the next position.
----------	--

---

**Return**

A page object.

**Head file reference**

fpd\_pageTempl.h: 97

**FPDPageGetObjectAt****Syntax**

```
FPD_PageObject FPDPageGetObjectAt (
    FPD Page page,
    FS\_POSITION pos
);
```



**Description**

Gets an object at specified position.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

pos	[In] Specifies the position of the page object.
-----	---

---

**Return**

A page object.

**Head file reference**

fpd\_pageTempl.h: 117

**FPDPageGetObjectByIndex****Syntax**

```
FPD_PageObject FPDPageGetObjectByIndex (
    FPD\_Page page,
    FS\_INT32 index
);
```

**Description**

Gets an object by a zero-based page object index.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

index	[In] Specifies the zero-based index of the page object.
-------	---

---

**Return**

A page object.

**Head file reference**

fpd\_pageTempl.h: 146

**FPDPageGetObjectIndex****Syntax**

```
FS_INT32 FPDPageGetObjectIndex (
    FPD\_Page page,
    FPD\_PageObject obj
);
```

**Description**

Gets the zero-based page object index in the object array.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

obj	[In] The input page object.
-----	-----------------------------

---

**Return**

The zero-based index of the page object.

**Head file reference**

fpd\_pageTempl.h: 136

**FPDPageGetPageAttr****Syntax**

```
FPD_Object FPDPageGetPageAttr (
    FPD\_Page page,
    FS\_LPSTR szName
);
```

**Description**

Gets an inheritable attribute value.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

szName	[In] The attribute(entry) name
--------	--------------------------------

---

**Return**

The attribute value

**Head file reference**

fpd\_pageTempl.h: 381

**FPDPageGetPageBBox****Syntax**

```
FS_FloatRect FPDPageGetPageBBox (
    FPD\_Page page
);
```

**Description**

Gets the page bounding box in user space.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

The page bounding box in user space.

**Head file reference**

fpd\_pageTempl.h: 363

**FPDPageGetPageHeight****Syntax**

```
FS_FLOAT FPDPageGetPageHeight (
    FPD\_Page page
);
```

**Description**

Gets the page height in user space.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

The page height in user space.

**Head file reference**

fpd\_pageTempl.h: 354

**FPDPageGetPageMatrix****Syntax**

```
FS_AffineMatrix FPDPageGetPageMatrix (
    FPD\_Page page
);
```

**Description**

Gets the page matrix.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

The page matrix.

**Head file reference**

fpd\_pageTempl.h: 372

**FPDPageGetPageText****Syntax**

```
void FPDPageGetPageText (
    FS\_ByteStringArray* outLines,
    FPD\_Document doc,
    FPD\_Object pageDic,
    FS\_INT32 iMinWidth,
    FS\_DWORD flags
);
```

**Description**

Extracts pure text from a page, in appearance order.

**Parameter**


---

outLines	[Out] It receives the text lines.
----------	-----------------------------------

---



---

doc	[In] The input PDF document.
-----	------------------------------

---



---

pageDic	[In] The input PDF page.
---------	--------------------------

---



---

iMinWidth	[In] The input minimum width of the text.
-----------	---

---



---

flags	[In] The input OS text extracting flags.
-------	--

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 418

**FPDPageGetPageText\_Uncode****Syntax**

```
void FPDPageGetPageText_Uncode (
    FS\_WideStringArray* outLines,
    FPD\_Document doc,
    FPD\_Object pageDic,
```

```
FS_INT32 iMinWidth,  
FS_DWORD flags  
);
```

**Description**

Extracts Unicode pure text from a page, in appearance order.

**Parameter**

outLines	[Out] It receives the text lines.
doc	[In] The input PDF document.
pageDic	[In] The input PDF page.
iMinWidth	[In] The input minimum width of the text.
flags	[In] The input OS text extracting flags.

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 431

**FPDPageGetPageWidth****Syntax**

```
FS_FLOAT FPDPageGetPageWidth (  
    FPD_Page page  
) ;
```

**Description**

Gets the page width in user space.

**Parameter**

page	[In] The input PDF page.
------	--------------------------

**Return**

The page width in user space.

**Head file reference**

fpd\_pageTempl.h: 345

## FPDPageGetParseState

### Syntax

```
FS_INT32 FPDPageGetParseState (
    FPD Page page
);
```

### Description

Gets the current parsing state.

### Parameter

---

page	[In] The input PDF page.
------	--------------------------

---

### Return

The current parsing state:

- FPD\_CONTENT\_NOT\_PARSED
- FPD\_CONTENT\_PARSING
- FPD\_CONTENT\_PARSED

### Head file reference

fpd\_pageTempl.h: 49

## FPDPageGetPrevObject

### Syntax

```
FPD_PageObject FPDPageGetPrevObject (
    FPD Page page,
    FS\_POSITION* inOutPos
);
```

### Description

Gets the current object and moves to previous position

### Parameter

---

page	[In] The input PDF page.
------	--------------------------

---

inOutPos	[In/Out] The input current position, and receives the previous position.
----------	--

---

### Return

A page object.

**Head file reference**

fpd\_pageTempl.h: 107

**FPDPageGetRenderCache****Syntax**

```
FPD_PageRenderCache FPDPageGetRenderCache (
    FPD\_Page page
);
```

**Description**

Gets render cache. for RENDER module.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

The render cache. for RENDER module.

**Head file reference**

fpd\_pageTempl.h: 391

**FPDPageGetResourcesDictionary****Syntax**

```
FPD_Object FPDPageGetResourcesDictionary (
    FPD\_Page page
);
```

**Description**

Gets the PDF page resources dictionary.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

The PDF page resources dictionary.

**Head file reference**

fpd\_pageTempl.h: 488

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.2.2](#)

## FPDPageInsertObject

### Syntax

```
FS_POSITION FPDPageInsertObject (
    FPD Page page,
    FS\_POSITION posInsertAfter,
    FPD PageObject newObj
);
```

### Description

Inserts a page object. To insert before all objects, use NULL for *posInsertAfter*.

### Parameter

---

page	[In] The input PDF page.
------	--------------------------

---

posInsertAfter	[In] Specifies the position to insert after.
----------------	--

---

newObj	[In] The input new page object.
--------	---------------------------------

---

### Return

The position of inserted page object

### Head file reference

fpd\_pageTempl.h: 167

## FPDPageIsParsed

### Syntax

```
FS_BOOL FPDPageIsParsed (
    FPD Page page
);
```

### Description

Checks whether the content has been parsed into page objects.

### Parameter

---

page	[In] The input PDF page.
------	--------------------------

---

### Return

[TRUE](#) if the content has been parsed into page objects, otherwise [FALSE](#).

### Head file reference

---

fpd\_pageTempl.h: 61

## FPDPageLoad

### Syntax

```
void FPDPageLoad (
    FPD Page page,
    FPD Document doc,
    FPD Object pageDict,
    FS BOOL bPageCache
);
```

### Description

Constructs a page. For saving memory, the page caching feature can be disabled, then images and masks used in page rendering won't be cached. Of course this will affect the speed.

### Parameter

page	[In] The input PDF page.
doc	[In] The PDF document
pageDict	[In] The page dictionary.
bPageCache	[In] Whether images and masks used in page rendering will be cached or not.

### Return

void

### Head file reference

fpd\_pageTempl.h: 296

## FPDPageMoveObject

### Syntax

```
FS_POSITION FPDPageMoveObject (
    FPD Page page,
    FS POSITION pos,
    FS POSITION newPosAfter
);
```

### Description

Moves a page object from a position to another position.

**Parameter**

page	[In] The input PDF page.
pos	[In] The original position of the page object.
newPosAfter	[In] The new position to move after.

**Return**

The new position of the page object.

**Head file reference**

fpd\_pageTempl.h: 188

**FPDPageNew****Syntax**

```
FPD_Page FPDPageNew (void );
```

**Description**

Creates a new empty PDF page.

**Return**

A new empty PDF page.

**Head file reference**

fpd\_pageTempl.h: 21

**FPDPageNewContentGenerator****Syntax**

```
FPD_ContentGenerator FPDPageNewContentGenerator (  
    FPD_Page page  
) ;
```

**Description**

Creates the PDF page content generator.

**Parameter**

page	[In] The input PDF page.
------	--------------------------

**Return**

The PDF page content generator.

**Head file reference**

fpd\_pageTempl.h: 444

**Related method**[FPDPageDestroyContentGenerator](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)**FPDPageParseContent****Syntax**

```
void FPDPageParseContent (
    FPD\_Page page,
    FPD\_ParseOptions options
);
```

**Description**

Parses all contents.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

options	[In] The parser options.
---------	--------------------------

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 320

**FPDPageRealizeResource****Syntax**

```
void FPDPageRealizeResource (
    FPD\_Page page,
    FPD\_Object pageResObj,
    FS\_MapPtrToPtr objMapping,
    FS\_LPSTR szType,
    FPD\_Object* outResObj,
    FS\_BytString* outResName
);
```

**Description**

Adds a resource to current object list. Returns the resource name.

**Parameter**


---

page	[In] The input PDF page.
pageResObj	[In] The input resource object
objMapping	[In] The input object mapping from object number to object pointer.
szType	[In] The resource type name.
outResObj	[Out] It receives the result resource object.
outResName	[Out] It receives the resource name.

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 257

**Note:** Will try to use an existing resource first, if available. Caller should not release the result resource object, which may be same as the input object. The input object can be an external object (which comes from another document, or archive), in this case, the object mapping should be specified.

**FPDPageRemoveObject****Syntax**

```
void FPDPageRemoveObject (
    FPD Page page,
    FS POSITION pos
);
```

**Description**

Removes a page object.

**Parameter**


---

page	[In] The input PDF page.
pos	[In] Specifies the position of the page object to be removed.

---

**Return**

**Head file reference**

fpd\_pageTempl.h: 178

**FPDPageReplaceObject****Syntax**

```
void FPDPageReplaceObject (
    FPD Page page,
    FS POSITION pos,
    FPD PageObject newObj
);
```

**Description**

Replaces a page object with a new page object.

**Parameter**

---

page	[In] The input PDF page.
pos	[In] Specifies the position of the page object to be replaced.
newObj	[In] The input new page object.

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 156

**FPDPageSetResourcesDictionary****Syntax**

```
void FPDPageSetResourcesDictionary (
    FPD Page page,
    FPD Object resourcesDict
);
```

**Description**

Sets the PDF page resources dictionary.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

---

resourcesDict	[In] The input PDF page resources dictionary.
---------------	---

---

**Return**

void.

**Head file reference**

fpd\_pageTempl.h: 499

**Related method****Since**[SDK\\_LATEEST\\_VERSION > 7.2.2](#)**FPDPageStartGenerateContent****Syntax**

```
FS_BOOL FPDPageStartGenerateContent (
    FPD\_ContentGenerator generator,
    FS\_FileStream fileStream
);
```

**Description**

Starts to generate the PDF content progressively.

**Parameter**


---

generator	[In] The input PDF content generator.
-----------	---------------------------------------

---

fileStream	[In] The input file stream object. Sets it NULL as default.
------------	---

---

**Return**

TRUE for success, otherwise not.

**Head file reference**

fpd\_pageTempl.h: 465

**Related method**[FPDPageContinueGenerateContent](#)**Since**[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)**FPDPageStartParse****Syntax**

```
void FPDPageStartParse (
    FPD\_Page page,
```

```
FPD_ParseOptions options  
);
```

**Description**

Start parsing the page. If pausing is enabled, application should check current parsing state after this function returns. If parsing not finished, ContinueParse() should be called.

**Parameter**

---

page	[In] The input PDF page.
options	[In] The parser options.

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 309

**FPDPageTransform****Syntax**

```
void FPDPageTransform (  
    FPD_Page page,  
    FS_AffineMatrix matrix  
) ;
```

**Description**

Transforms all objects.

**Parameter**

---

page	[In] The input PDF page.
matrix	[In] The input transformation matrix.

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 209

**FPD\_PageArchiveLoader**

[\*\*Return from Used by\*\*](#)**Description**

PDF page archive loader class. See [CFPDPageArchiveLoaderNew](#) , [CFPDPageArchiveLoaderDestroy](#) .

**Returned from**[\*\*FPDPageArchiveLoaderNew\*\*](#)**Used by**

[FPDPageArchiveLoaderDestroy](#)  
[FPDPageArchiveLoaderLoadClipPath](#)  
[FPDPageArchiveLoaderLoadColorState](#)  
[FPDPageArchiveLoaderLoadGeneralState](#)  
[FPDPageArchiveLoaderLoadGraphState](#)  
[FPDPageArchiveLoaderLoadObject](#)  
[FPDPageArchiveLoaderLoadTextState](#)

**Functions****Functions summary**[\*\*FPDPageArchiveLoaderDestroy\*\*](#)

Destroys the PDF page archive loader.

[\*\*FPDPageArchiveLoaderLoadClipPath\*\*](#)

Load or restore the clipping path.

[\*\*FPDPageArchiveLoaderLoadColorState\*\*](#)

Load or restore the color state.

[\*\*FPDPageArchiveLoaderLoadGeneralState\*\*](#)

Load or restore the general state.

[\*\*FPDPageArchiveLoaderLoadGraphState\*\*](#)

Load or restore the graph state.

[\*\*FPDPageArchiveLoaderLoadObject\*\*](#)

Loads an object from archive.

[\*\*FPDPageArchiveLoaderLoadTextState\*\*](#)

Load or restore the text state.

[\*\*FPDPageArchiveLoaderNew\*\*](#)

Constructs a loading archive.

**Functions detail**[\*\*FPDPageArchiveLoaderDestroy\*\*](#)**Syntax**

```
void FPDPageArchiveLoaderDestroy (
    FPD_PageArchiveLoader ar
);
```

**Description**

Destroys the PDF page archive loader.

**Parameter**

---

ar	[In] The input page archive loader.
----	-------------------------------------

---

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 383

**FPDPageArchiveLoaderLoadClipPath****Syntax**

```
void FPDPageArchiveLoaderLoadClipPath (
    FPD\_PageArchiveLoader ar,
    FPD\_ClipPath* clip_path
);
```

**Description**

Load or restore the clipping path.

**Parameter**

---

ar	[In] Ref to the input page archive loader.
----	--

---

clip_path	[Out] It receives the clipping path.
-----------	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 402

**FPDPageArchiveLoaderLoadColorState****Syntax**

```
void FPDPageArchiveLoaderLoadColorState (
    FPD\_PageArchiveLoader ar,
    FPD\_ColorState* color_state
);
```

**Description**

Load or restore the color state.

**Parameter**

---

ar	[In] Ref to the input page archive loader.
----	--

---

color_state	[Out] It receives the color state.
-------------	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 432

**FPDPageArchiveLoaderLoadGeneralState****Syntax**

```
void FPDPageArchiveLoaderLoadGeneralState (
    FPD\_PageArchiveLoader ar,
    FPD\_GeneralState* general_state
);
```

**Description**

Load or restore the general state.

**Parameter**

---

ar	[In] Ref to the input page archive loader.
----	--

---

general_state	[Out] It receives the general state.
---------------	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 442

**FPDPageArchiveLoaderLoadGraphState****Syntax**

```
void FPDPageArchiveLoaderLoadGraphState (
    FPD\_PageArchiveLoader ar,
    FPD\_GraphState* graph_state
);
```

**Description**

Load or restore the graph state.

**Parameter**

---

ar	[In] Ref to the input page archive loader.
graph_state	[Out] It receives the graph state.

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 412

**FPDPageArchiveLoaderLoadObject****Syntax**

```
void FPDPageArchiveLoaderLoadObject (
    FPD\_PageArchiveLoader ar,
    FPD\_PageObject* pObj
);
```

**Description**

Loads an object from archive.

**Parameter**

---

ar	[In] Ref to the input page archive loader.
pObj	[Out] It receives the loaded PDF object.

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 392

**FPDPageArchiveLoaderLoadTextState****Syntax**

```
void FPDPageArchiveLoaderLoadTextState (
    FPD\_PageArchiveLoader ar,
    FPD\_TextState* text_state
);
```

**Description**

Load or restore the text state.

**Parameter**


---

ar	[In] Ref to the input page archive loader.
text_state	[Out] It receives the text state.

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 422

**FPDPageArchiveLoaderNew****Syntax**

```
FPD_PageArchiveLoader FPDPageArchiveLoaderNew (
    FPD\_Page pPageObjs,
    FS\_LPCBYTE pData,
    FS\_DWORD dwSize
);
```

**Description**

Constructs a loading archive.

**Parameter**


---

pPageObjs	[In] The current page/form.
pData	[In] The input memory buffer.
dwSize	[In] The size of the input buffer.

---

**Return**

The PDF page archive loader.

**Head file reference**

fpd\_serialTempl.h: 372

**FPD\_PageArchiveSaver**

[Return from Used by](#)

## Description

PDF page archive saver class. See [CFPDPageArchiveSaverNew](#) , [CFPDPageArchiveSaverDestroy](#) .

### Returned from

[FPDPageArchiveSaverNew](#)

### Used by

[FPDPageArchiveSaverDestroy](#)  
[FPDPageArchiveSaverSaveClipPath](#)  
[FPDPageArchiveSaverSaveColorState](#)  
[FPDPageArchiveSaverSaveGeneralState](#)  
[FPDPageArchiveSaverSaveGraphState](#)  
[FPDPageArchiveSaverSavePageObject](#)  
[FPDPageArchiveSaverSaveTextState](#)

## Functions

### Functions summary

[FPDPageArchiveSaverDestroy](#)

Destroys the PDF page archive saver.

[FPDPageArchiveSaverNew](#)

Creates a PDF page archive saver.

[FPDPageArchiveSaverSaveClipPath](#)

Saves clipping path.

[FPDPageArchiveSaverSaveColorState](#)

Saves color state.

[FPDPageArchiveSaverSaveGeneralState](#)

Saves general state.

[FPDPageArchiveSaverSaveGraphState](#)

Saves graph state.

[FPDPageArchiveSaverSavePageObject](#)

Saves page object.

[FPDPageArchiveSaverSaveTextState](#)

Saves text state.

### Functions detail

[FPDPageArchiveSaverDestroy](#)

#### Syntax

```
void FPDPageArchiveSaverDestroy (  
    FPD\_PageArchiveSaver ar  
);
```

#### Description

Destroys the PDF page archive saver.

**Parameter**

---

ar	[In] The input page archive saver.
----	------------------------------------

---

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 295

**FPDPageArchiveSaverNew****Syntax**

```
FPD_PageArchiveSaver FPDPageArchiveSaverNew (
    FPD\_Page pPageObjs
);
```

**Description**

Creates a PDF page archive saver.

**Parameter**

---

pPageObjs	[In] The input page object list.
-----------	----------------------------------

---

**Return**

The PDF page archive saver.

**Head file reference**

fpd\_serialTempl.h: 286

**FPDPageArchiveSaverSaveClipPath****Syntax**

```
void FPDPageArchiveSaverSaveClipPath (
    FPD\_PageArchiveSaver ar,
    FPD\_ClipPath clip_path
);
```

**Description**

Saves clipping path.

**Parameter**

---

ar	[In] Ref to the output page archive saver.
----	--

---

---

clip_path	[In] The input clipping path.
-----------	-------------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 314

**FPDPageArchiveSaverSaveColorState****Syntax**

```
void FPDPageArchiveSaverSaveColorState (
    FPD\_PageArchiveSaver ar,
    FPD\_ColorState color_state
);
```

**Description**

Saves color state.

**Parameter**

---

ar	[In] Ref to the output page archive saver.
----	--

---

---

color_state	[In] The input color state.
-------------	-----------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 344

**FPDPageArchiveSaverSaveGeneralState****Syntax**

```
void FPDPageArchiveSaverSaveGeneralState (
    FPD\_PageArchiveSaver ar,
    FPD\_GeneralState general_state
);
```

**Description**

Saves general state.

**Parameter**

---

ar	[In] Ref to the output page archive saver.
----	--

---

---

general_state	[In] The input general state.
---------------	-------------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 354

**FPDPageArchiveSaverSaveGraphState****Syntax**

```
void FPDPageArchiveSaverSaveGraphState (
    FPD\_PageArchiveSaver ar,
    FPD\_GraphState graph_state
);
```

**Description**

Saves graph state.

**Parameter**

---

ar	[In] Ref to the output page archive saver.
----	--

---

---

graph_state	[In] The input graph state.
-------------	-----------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 324

**FPDPageArchiveSaverSavePageObject****Syntax**

```
void FPDPageArchiveSaverSavePageObject (
    FPD\_PageArchiveSaver ar,
    FPD\_PageObject pObj
);
```

**Description**

Saves page object.

**Parameter**

---

ar	[In] Ref to the output page archive saver.
----	--

---

---

pObj	[In] The input page object.
------	-----------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 304

**FPDPageArchiveSaverSaveTextState****Syntax**

```
void FPDPageArchiveSaverSaveTextState (
    FPD\_PageArchiveSaver ar,
    FPD\_TextState text_state
);
```

**Description**

Saves text state.

**Parameter**

---

ar	[In] Ref to the output page archive saver.
----	--

---

---

text_state	[In] The input text state.
------------	----------------------------

---

**Return**

void

**Head file reference**

fpd\_serialTempl.h: 334

# FPD\_PageObject

**[Return from Used by](#)****Description**

A [FPD\\_PageObject](#) is the abstract superclass of page objects. You can find the type of any object with the [FPDPageObjectGetType\(\)](#) method.

**Returned from**

[FPDClipPathGetText](#)  
[FPDFFormGetNextObject](#)  
[FPDFFormGetObjectAt](#)  
[FPDFFormGetObjectByIndex](#)  
[FPDFFormGetPrevObject](#)

[FPDFFormObjectNew](#)  
[FPDImageObjectNew](#)  
[FPDInlineImagesNew](#)  
[FPDPPageGetNextObject](#)  
[FPDPPageGetObjectAt](#)  
[FPDPPageGetObjectByIndex](#)  
[FPDPPageGetPrevObject](#)  
[FPDPathObjectNew](#)  
[FPDShadingObjectNew](#)  
[FPDTextObjectNew](#)  
[FPDPPageObjectClone](#)  
[FPDPPageObjectNew](#)

### Used by

[FPDClipPathAppendTexts](#)  
[FPDFFormGetObjectIndex](#)  
[FPDFFormInsertObject](#)  
[FPDFFormReplaceObject](#)  
[FPDFFormObjectCalcBoundingBox](#)  
[FPDFFormObjectDestroy](#)  
[FPDFFormObjectGetForm](#)  
[FPDFFormObjectGetTransformMatrix](#)  
[FPDFFormObjectSetForm](#)  
[FPDFFormObjectSetTransformMatrix](#)  
[FPDFFormObjectTransform](#)  
[FPDImageObjectCalcBoundingBox](#)  
[FPDImageObjectDestroy](#)  
[FPDImageObjectGetImage](#)  
[FPDImageObjectGetTransformMatrix](#)  
[FPDImageObjectSetImage](#)  
[FPDImageObjectSetTransformMatrix](#)  
[FPDImageObjectTransform](#)  
[FPDInlineImagesAddMatrix](#)  
[FPDInlineImagesCountMatrix](#)  
[FPDInlineImagesDestroy](#)  
[FPDInlineImagesGetMatrix](#)  
[FPDInlineImagesGetStream](#)  
[FPDInlineImagesSetStream](#)  
[FPDPPageGetObjectIndex](#)  
[FPDPPageInsertObject](#)  
[FPDPPageReplaceObject](#)  
[FPDPPageArchiveLoaderLoadObject](#)  
[FPDPPageArchiveSaverSavePageObject](#)  
[FPDPathObjectCalcBoundingBox](#)  
[FPDPathObjectDestroy](#)  
[FPDPathObjectGetFillMode](#)  
[FPDPathObjectGetPath](#)  
[FPDPathObjectGetTransformMatrix](#)  
[FPDPathObjectIsStrokeMode](#)  
[FPDPathObjectSetFillMode](#)  
[FPDPathObjectSetGraphState](#)  
[FPDPathObjectSetStrokeMode](#)  
[FPDPathObjectSetTransformMatrix](#)  
[FPDPathObjectTransform](#)  
[FPDRenderDeviceDrawType3Text](#)

[FPDShadingObjectCalcBoundingBox](#)  
[FPDShadingObjectDestroy](#)  
[FPDShadingObjectGetPage](#)  
[FPDShadingObjectGetShadingPattern](#)  
[FPDShadingObjectGetTransformMatrix](#)  
[FPDShadingObjectSetPage](#)  
[FPDShadingObjectSetShadingPattern](#)  
[FPDShadingObjectSetTransformMatrix](#)  
[FPDShadingObjectTransform](#)  
[FPDTextObjectCalcCharPos](#)  
[FPDTextObjectCountChars](#)  
[FPDTextObjectCountItems](#)  
[FPDTextObjectDestroy](#)  
[FPDTextObjectGetCharInfo](#)  
[FPDTextObjectGetData](#)  
[FPDTextObjectGetFont](#)  
[FPDTextObjectGetFontSize](#)  
[FPDTextObjectGetItemInfo](#)  
[FPDTextObjectGetPosX](#)  
[FPDTextObjectGetPosY](#)  
[FPDTextObjectGetTextMatrix](#)  
[FPDTextObjectRecalcPositionData](#)  
[FPDTextObjectSetData](#)  
[FPDTextObjectSetEmpty](#)  
[FPDTextObjectSetPosition](#)  
[FPDTextObjectSetText](#)  
[FPDTextObjectSetText2](#)  
[FPDTextObjectSetText3](#)  
[FPDTextObjectSetTextState](#)  
[FPDTextObjectTransform](#)  
[FPDPageObjectAppendClipPath](#)  
[FPDPageObjectClone](#)  
[FPDPageObjectCopy](#)  
[FPDPageObjectCopyClipPath](#)  
[FPDPageObjectCopyStates](#)  
[FPDPageObjectDefaultStates](#)  
[FPDPageObjectDestroy](#)  
[FPDPageObjectGetBBox](#)  
[FPDPageObjectGetClipPath](#)  
[FPDPageObjectGetColorState](#)  
[FPDPageObjectGetContentMark](#)  
[FPDPageObjectGetContentMark2](#)  
[FPDPageObjectGetGeneralState](#)  
[FPDPageObjectGetGraphState](#)  
[FPDPageObjectGetOriginalBBox](#)  
[FPDPageObjectGetTextState](#)  
[FPDPageObjectGetType](#)  
[FPDPageObjectHasClipPath](#)  
[FPDPageObjectRemoveClipPath](#)  
[FPDPageObjectSetColorState](#)  
[FPDPageObjectSetGeneralState](#)  
[FPDPageObjectSetGraphState](#)  
[FPDPageObjectSetTextState](#)  
[FPDPageObjectTransformClipPath](#)

## Definitions

## Definitions summary

### [FPD\\_PAGEOBJ\\_FORM](#)

form object.

### [FPD\\_PAGEOBJ\\_IMAGE](#)

image object.

### [FPD\\_PAGEOBJ\\_INLINES](#)

inline image object. Inline image section. FOR EMBEDDED SYSTEM ONLY

### [FPD\\_PAGEOBJ\\_PATH](#)

path object.

### [FPD\\_PAGEOBJ\\_SHADING](#)

shading object.

### [FPD\\_PAGEOBJ\\_TEXT](#)

text object.

### [FPDPT\\_BEZIERTO](#)

Specifies that this point is a control point or ending point for a Bézier spline.

### [FPDPT\\_CLOSEFIGURE](#)

Specifies that the figure is automatically closed after the [FPDPT\\_LINETO](#) or [FPDPT\\_BEZIERTO](#) or this point is done. A line is drawn from this point to the most recent #FPDPT\_MOVETO point.

### [FPDPT\\_LINETO](#)

Specifies that a line is to be drawn from the current position to this point, which then becomes the new current position.

### [FPDPT\\_MOVETO](#)

Specifies that this point starts a figure. This point becomes the new current position.

### [FPDPT\\_TYPE](#)

Denotes point type mask. Point types are:

#FPDPT\_MOVETO/#FPDPT\_LINETO/#FPDPT\_BEZIERTO.

## Definitions detail

### FPD\_PAGEOBJ\_FORM

#### Syntax

```
#define FPD_PAGEOBJ_FORM 5
```

#### Description

form object.

#### Group

[FPDPageObjectTypes](#)

#### Head file reference

fpd\_pageobjExpT.h: 218

### FPD\_PAGEOBJ\_IMAGE

#### Syntax

```
#define FPD_PAGEOBJ_IMAGE 3
```

**Description**

image object.

**Group**

[FPDPageObjectTypes](#)

**Head file reference**

fpd\_pageobjExpT.h: 214

## FPD\_PAGEOBJ\_INLINES

**Syntax**

#define FPD\_PAGEOBJ\_INLINES 6

**Description**

inline image object. Inline image section. FOR EMBEDDED SYSTEM ONLY

**Group**

[FPDPageObjectTypes](#)

**Head file reference**

fpd\_pageobjExpT.h: 220

## FPD\_PAGEOBJ\_PATH

**Syntax**

#define FPD\_PAGEOBJ\_PATH 2

**Description**

path object.

**Group**

[FPDPageObjectTypes](#)

**Head file reference**

fpd\_pageobjExpT.h: 212

## FPD\_PAGEOBJ\_SHADING

**Syntax**

#define FPD\_PAGEOBJ\_SHADING 4

**Description**

shading object.

**Group**

[FPPDPageObjectTypes](#)**Head file reference**

fpd\_pageobjExpT.h: 216

**FPD\_PAGEOBJ\_TEXT****Syntax**

#define FPD\_PAGEOBJ\_TEXT 1

**Description**

text object.

**Group**[FPPDPageObjectTypes](#)**Head file reference**

fpd\_pageobjExpT.h: 210

**FPDPT\_BEZIERTO****Syntax**

#define FPDPT\_BEZIERTO 0x04

**Description**

Specifies that this point is a control point or ending point for a Bézier spline.

**Group**[FPPDPathPointFlags](#)**Head file reference**

fpd\_pageobjExpT.h: 194

**FPDPT\_CLOSEFIGURE****Syntax**

#define FPDPT\_CLOSEFIGURE 0x01

**Description**Specifies that the figure is automatically closed after the [FPDPT\\_LINETO](#) or [FPDPT\\_BEZIERTO](#) or this point is done. A line is drawn from this point to the most recent #FPDPT\_MOVETO point.**Group**[FPPDPathPointFlags](#)**Head file reference**

fpd\_pageobjExpT.h: 190



## FPDPT\_LINETO

### Syntax

```
#define FPDPT_LINETO 0x02
```

### Description

Specifies that a line is to be drawn from the current position to this point, which then becomes the new current position.

### Group

[FPDPathPointFlags](#)

### Head file reference

fpd\_pageobjExpT.h: 192

## FPDPT\_MOVETO

### Syntax

```
#define FPDPT_MOVETO 0x06
```

### Description

Specifies that this point starts a figure. This point becomes the new current position.

### Group

[FPDPathPointFlags](#)

### Head file reference

fpd\_pageobjExpT.h: 196

## FPDPT\_TYPE

### Syntax

```
#define FPDPT_TYPE 0x06
```

### Description

Denotes point type mask. Point types are:  
#FPDPT\_MOVETO/#FPDPT\_LINETO/#FPDPT\_BEZIERTO.

### Group

[FPDPathPointFlags](#)

### Head file reference

fpd\_pageobjExpT.h: 198

## Structures

## Structures summary

A data structure for text object item.

### Structs detail

#### Syntax

```
typedef struct FPD_TextObjectItem{  
    FS_DWORD m_CharCode,  
    FS_FLOAT m-OriginX,  
    FS_FLOAT m-OriginY  
};
```

#### Description

A data structure for text object item.

#### Head file reference

fpd\_pageobjExpT.h: 227

##### m\_CharCode

The character code. -1 (0xffffffff) for spacing.

##### m-OriginX

X origin in text space, or the spacing.

##### m-OriginY

Y origin in text space.

## Functions

### Functions summary

#### [FPDPageObjectAppendClipPath](#)

Appends a clipping path.

#### [FPDPageObjectClone](#)

Clones a page object.

#### [FPDPageObjectCopy](#)

Copies from another page object.

#### [FPDPageObjectCopyClipPath](#)

Copies clipping path from another object.

#### [FPDPageObjectCopyStates](#)

Copies from another graphic states.

#### [FPDPageObjectDefaultStates](#)

Sets all graphic states to default.

#### [FPDPageObjectDestroy](#)

Destroys the PDF page object.

#### [FPDPageObjectGetBBox](#)

Gets the bounding box of the page object, optionally with a transformation matrix.

#### [FPDPageObjectGetClipPath](#)

Gets the clip path state.

**[FPDPageObjectGetColorState](#)**

Gets the color state. For texts, graphs and uncolored images.

**[FPDPageObjectGetContentMark](#)**

Gets the content mark.

**[FPDPageObjectGetContentMark2](#)**

Gets the content mark.

**[FPDPageObjectGetGeneralState](#)**

Gets the general state. For all objects.

**[FPDPageObjectGetGraphState](#)**

Gets the graph state. For graphs and type3 font or stroke texts.

**[FPDPageObjectGetOriginalBBox](#)**

Gets the original bounding box of the page object.

**[FPDPageObjectGetTextState](#)**

Gets the text state. For texts only.

**[FPDPageObjectGetType](#)**

The page object type.

**[FPDPageObjectHasClipPath](#)**

Checks whether the page object has the clip path or not.

**[FPDPageObjectNew](#)**

Creates a new empty PDF page object.

**[FPDPageObjectRemoveClipPath](#)**

Removes clipping path of the object.

**[FPDPageObjectSetColorState](#)**

Sets the color state.

**[FPDPageObjectSetGeneralState](#)**

Sets the general state.

**[FPDPageObjectSetGraphState](#)**

Sets the graph state.

**[FPDPageObjectSetTextState](#)**

Sets the text state.

**[FPDPageObjectTransformClipPath](#)**

Transforms the clip path. Rotate, shear, or move clip path.

## Functions detail

### **FPDPageObjectAppendClipPath**

#### Syntax

```
void FPDPageObjectAppendClipPath (
    FPD\_PageObject pageObj,
    FPD\_Path path,
    FS\_INT32 type,
    FS\_BOOL bAutoMerge
);
```

#### Description

Appends a clipping path.

#### Parameter

pageObj	[In] The input PDF page object.
path	[In] The input clipping path.
type	[In] The clip type of the input clipping path.
bAutoMerge	[In] Whether to merge the clipping path automatically.

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1056

**FPDPageObjectClone****Syntax**

```
FPD_PageObject FPDPageObjectClone (
    FPD\_PageObject pageObj
);
```

**Description**

Clones a page object.

**Parameter**

pageObj	[In] The source object.
---------	-------------------------

**Return**

The cloned object.

**Head file reference**

fpd\_pageobjTempl.h: 1028

**FPDPageObjectCopy****Syntax**

```
void FPDPageObjectCopy (
    FPD\_PageObject pageObj,
    const FPD\_PageObject srcObject
);
```

**Description**

Copies from another page object.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

srcObject	[In] The source page object.
-----------	------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1037

**FPDPageObjectCopyClipPath****Syntax**

```
void FPDPageObjectCopyClipPath (
    FPD\_PageObject pageObj,
    FPD\_PageObject srcObj
);
```

**Description**

Copies clipping path from another object.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

srcObj	[In] The source page object.
--------	------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1068

**FPDPageObjectCopyStates****Syntax**

```
void FPDPageObjectCopyStates (
    FPD\_PageObject pageObj,
    const FPD\_PageObject src
);
```

**Description**

Copies from another graphic states.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

src	[In] The input graphic states.
-----	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1189

**FPDPageObjectDefaultStates****Syntax**

```
void FPDPageObjectDefaultStates (
    FPD\_PageObject pageObj
);
```

**Description**

Sets all graphic states to default.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1180

**FPDPageObjectDestroy****Syntax**

```
void FPDPageObjectDestroy (
    FPD\_PageObject pageObj
);
```

**Description**

Destroys the PDF page object.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1019

**FPDPageObjectGetBBox****Syntax**

```
FS_Rect FPDPageObjectGetBBox (
    FPD\_PageObject pageObj,
    FS\_AffineMatrix* pMatrix
);
```

**Description**

Gets the bounding box of the page object, optionally with a transformation matrix.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

---

pMatrix	[In] The input transformation matrix.
---------	---------------------------------------

---

**Return**

The bounding box of the page object.

**Head file reference**

fpd\_pageobjTempl.h: 1098

**FPDPageObjectGetClipPath****Syntax**

```
FPD_ClipPath FPDPageObjectGetClipPath (
    FPD\_PageObject pageObj
);
```

**Description**

Gets the clip path state.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

**Return**

The clip path state.

**Head file reference**

fpd\_pageobjTempl.h: 1126

**FPDPageObjectGetColorState****Syntax**

```
FPD_ColorState FPDPageObjectGetColorState (
    FPD\_PageObject pageObj
);
```

**Description**

Gets the color state. For texts, graphs and uncolored images.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

**Return**

The color state for texts, graphs and uncolored images.

**Head file reference**

fpd\_pageobjTempl.h: 1144

**FPDPageObjectGetContentMark****Syntax**

```
FPD_ContentMark FPDPageObjectGetContentMark (
    FPD\_PageObject pageObj
);
```

**Description**

Gets the content mark.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

**Return**

The content mark. For all objects.

**Head file reference**

fpd\_pageobjTempl.h: 1171

**FPDPageObjectGetContentMark2**

**Syntax**

```
FPD_ContentMark FPDPageObjectGetContentMark2 (
    FPD\_PageObject pageObj,
    FS\_BOOL bGetModifiableCopy
);
```

**Description**

Gets the content mark.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

bGetModifiableCopy	[In] Get a modifiable copy of the object. If the reference was refer to null, then a new object will be created. The returned pointer can be used to alter the object content.
--------------------	--

---

**Return**

The content mark. For all objects.

**Head file reference**

fpd\_pageobjTempl.h: 1239

**Since**

[SDK\\_LATEEST\\_VERSION > 8.3.1](#)

**FPDPageObjectGetGeneralState****Syntax**

```
FPD_GeneralState FPDPageObjectGetGeneralState (
    FPD\_PageObject pageObj
);
```

**Description**

Gets the general state. For all objects.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

**Return**

The general state for all objects.

**Head file reference**

fpd\_pageobjTempl.h: 1162

## FPDPageObjectGetGraphState

### Syntax

```
FPD_GraphState FPDPageObjectGetGraphState (
    FPD\_PageObject pageObj
);
```

### Description

Gets the graph state. For graphs and type3 font or stroke texts.

### Parameter

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

### Return

The graph state for graphs and type3 font or stroke texts.

### Head file reference

fpd\_pageobjTempl.h: 1135

## FPDPageObjectGetOriginalBBox

### Syntax

```
FS_FloatRect FPDPageObjectGetOriginalBBox (
    FPD\_PageObject pageObj
);
```

### Description

Gets the original bounding box of the page object.

### Parameter

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

### Return

The original bounding box of the page object

### Head file reference

fpd\_pageobjTempl.h: 1108

## FPDPageObjectGetTextState

### Syntax

```
FPD_TextState FPDPageObjectGetTextState (
    FPD\_PageObject pageObj
);
```

**Description**

Gets the text state. For texts only.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

**Return**

The text state for texts only.

**Head file reference**

fpd\_pageobjTempl.h: 1153

**FPDPageObjectGetType****Syntax**

```
FS_INT32 FPDPageObjectGetType (
    FPD\_PageObject pageObj
);
```

**Description**

The page object type.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

**Return**

Gets the page object type.

**Head file reference**

fpd\_pageobjTempl.h: 1117

**FPDPageObjectHasClipPath****Syntax**

```
FS_BOOL FPDPageObjectHasClipPath (
    FPD\_PageObject pageObj
);
```

**Description**

Checks whether the page object has the clip path or not.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

**Return**

[TRUE](#) if the page object has the clip path, otherwise not.

**Head file reference**

fpd\_pageobjTempl.h: 1229

**Since**

[SDK LATEST VERSION > 2.0](#)

**FPDPageObjectNew****Syntax**

```
FPD_PageObject FPDPageObjectNew (
    FS\_INT32 type
);
```

**Description**

Creates a new empty PDF page object.

**Parameter**

---

type	[In] The input page object type.
------	----------------------------------

---

**Return**

A page object.

**Head file reference**

fpd\_pageobjTempl.h: 1010

**FPDPageObjectRemoveClipPath****Syntax**

```
void FPDPageObjectRemoveClipPath (
    FPD\_PageObject pageObj
);
```

**Description**

Removes clipping path of the object.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1047

**FPDPageObjectSetColorState****Syntax**

```
void FPDPageObjectSetColorState (
    FPD\_PageObject pageObj,
    FPD\_ColorState state
);
```

**Description**

Sets the color state.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

state	[In] The new color state.
-------	---------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1088

**FPDPageObjectSetGeneralState****Syntax**

```
void FPDPageObjectSetGeneralState (
    FPD\_PageObject pageObj,
    FPD\_GeneralState genState
);
```

**Description**

Sets the general state.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

genState	[In] The general state to be set.
----------	-----------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1219

**FPDPageObjectSetGraphState****Syntax**

```
void FPDPageObjectSetGraphState (
    FPD\_PageObject pageObj,
    FPD\_GraphState graphState
);
```

**Description**

Sets the graph state.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

graphState	[In] The graphic state to be set.
------------	-----------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1199

**FPDPageObjectSetTextState****Syntax**

```
void FPDPageObjectSetTextState (
    FPD\_PageObject pageObj,
    FPD\_TextState textState
);
```

**Description**

Sets the text state.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

textState	[In] The text state to be set.
-----------	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1209

## FPDPageObjectTransformClipPath

**Syntax**

```
void FPDPageObjectTransformClipPath (
    FPD\_PageObject pageObj,
    FS\_AffineMatrix matrix
);
```

**Description**

Transforms the clip path. Rotate, shear, or move clip path.

**Parameter**

---

pageObj	[In] The input PDF page object.
---------	---------------------------------

---

matrix	[In] The matrix used to transform.
--------	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1078

## FPD\_PageRenderCache

**[Return from](#) [Used by](#)****Description**

Store all dispensable items used when rendering a page. See [FPD\\_BackgroundDraw](#) Note:  
Currently we have image cache only.

**Returned from**

[FPDPageGetRenderCache](#)  
[FPDRenderContextGetPageCache](#)  
[FPDPageRenderCacheNew](#)

**Used by**

[FPDRenderContextNew2](#)  
[FPDPageRenderCacheClearAll](#)  
[FPDPageRenderCacheClearImageData](#)  
[FPDPageRenderCacheDestroy](#)  
[FPDPageRenderCacheEstimateSize](#)  
[FPDPageRenderCacheGetCachedBitmap](#)  
[FPDPageRenderCacheResetBitmap](#)

## Functions

### Functions summary

#### [FPDPageRenderCacheClearAll](#)

Clears all items in the cache.

#### [FPDPageRenderCacheClearImageData](#)

Clears image data.

#### [FPDPageRenderCacheDestroy](#)

Destroys the page rendering cache object.

#### [FPDPageRenderCacheEstimateSize](#)

Gets estimated size of the total cache.

#### [FPDPageRenderCacheGetCachedBitmap](#)

Gets cached items

#### [FPDPageRenderCacheNew](#)

Creates a new empty page rendering cache object.

#### [FPDPageRenderCacheResetBitmap](#)

Resets the image cache or force the cache to be expired

### Functions detail

#### FPDPageRenderCacheClearAll

##### Syntax

```
void FPDPageRenderCacheClearAll (  
    FPD PageRenderCache cache  
) ;
```

##### Description

Clears all items in the cache.

##### Parameter

---

cache	[In] The input page rendering cache object.
-------	---

---

##### Return

void

##### Head file reference

fpd\_renderTempl.h: 908

## FPDPageRenderCacheClearImageData

### Syntax

```
void FPDPageRenderCacheClearImageData (
    FPD\_PageRenderCache cache
);
```

### Description

Clears image data.

### Parameter

---

cache	[In] The input page rendering cache object.
-------	---

---

### Return

void

### Head file reference

fpd\_renderTempl.h: 917

## FPDPageRenderCacheDestroy

### Syntax

```
void FPDPageRenderCacheDestroy (
    FPD\_PageRenderCache cache
);
```

### Description

Destroys the page rendering cache object.

### Parameter

---

cache	[In] The input page rendering cache object.
-------	---

---

### Return

void

### Head file reference

fpd\_renderTempl.h: 899

## FPDPageRenderCacheEstimateSize

### Syntax

```
FS_DWORD FPDPageRenderCacheEstimateSize (
    FPD\_PageRenderCache cache
);
```

**Description**

Gets estimated size of the total cache.

**Parameter**

---

cache	[In] The input page rendering cache object.
-------	---

---

**Return**

Estimated size of the total cache.

**Head file reference**

fpd\_renderTempl.h: 926

**Note:** The application can use some strategy for dispense page caches according the memory size they occupy.

**FPDPageRenderCacheGetCachedBitmap****Syntax**

```
void FPDPageRenderCacheGetCachedBitmap (
    FPD\_PageRenderCache cache,
    FPD\_Object stream,
    FS\_DIBitmap* outBitmap,
    FS\_DIBitmap* outMask,
    FS\_DWORD* outMatteColor
);
```

**Description**

Gets cached items

**Parameter**

---

cache	[In] The input page rendering cache object.
-------	---

---

---

stream	[In] The stream of the bitmap.
--------	--------------------------------

---

---

outBitmap	[Out] It retrieves the bitmap.
-----------	--------------------------------

---

---

outMask	[Out] It retrieves the mask.
---------	------------------------------

---

---

outMatteColor	[Out] It retrieves the matte color.
---------------	-------------------------------------

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 935

**FPDPageRenderCacheNew****Syntax**

```
FPD_PageRenderCache FPDPageRenderCacheNew (
    FPD Page page
);
```

**Description**

Creates a new empty page rendering cache object.

**Parameter**

---

page	[In] The input PDF page.
------	--------------------------

---

**Return**

A new empty page rendering cache object.

**Head file reference**

fpd\_renderTempl.h: 890

**FPDPageRenderCacheResetBitmap****Syntax**

```
void FPDPageRenderCacheResetBitmap (
    FPD PageRenderCache cache,
    FPD Object stream,
    const FS\_DIBitmap bitmap
);
```

**Description**

Resets the image cache or force the cache to be expired

**Parameter**

---

cache	[In] The input page rendering cache object.
-------	---

---

stream	[In] The stream of the bitmap.
--------	--------------------------------

---

bitmap	[In] Input the bitmap.
--------	------------------------

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 948

## FPD\_ParseOptions

[Return from Used by](#)

### Description

The page parsing options.

### Returned from

[FPDParseOptionsNew](#)

### Used by

[FPDFFormParseContent](#)  
[FPDFFormStartParse](#)  
[FPDPageParseContent](#)  
[FPDPageStartParse](#)  
[FPDParseOptionsDestroy](#)  
[FPDParseOptionsSetFormGenerateFlag](#)  
[FPDParseOptionsSetInlineImageDecodeFlag](#)  
[FPDParseOptionsSetMarkedContentLoadFlag](#)  
[FPDParseOptionsSetTextFlag](#)

### Functions

#### Functions summary

[FPDParseOptionsDestroy](#)

Destroys the page parsing options object.

[FPDParseOptionsNew](#)

Creates a new empty page parsing options object.

[FPDParseOptionsSetFormGenerateFlag](#)

Sets whether Generate FPD\_FormObject for form or not.

[FPDParseOptionsSetInlineImageDecodeFlag](#)

Sets whether decode inline image for better performance, This should be disabled for PDF Editor, to keep the result file size smaller.

[FPDParseOptionsSetMarkedContentLoadFlag](#)

Sets whether load marked content (including foxit tag) information or not.

[FPDParseOptionsSetTextFlag](#)

Sets whether only text object is parsed into object list or not.

#### Functions detail

[FPDParseOptionsDestroy](#)

**Syntax**

```
void FPDParseOptionsDestroy (
    FPD ParseOptions opt
);
```

**Description**

Destroys the page parsing options object.

**Parameter**


---

opt	[In] The input page parsing options object.
-----	---

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 532

**FPDParseOptionsNew****Syntax**

```
FPD_ParseOptions FPDParseOptionsNew (
    FS\_BOOL bTextOnly,
    FS\_BOOL bMarkedContent,
    FS\_BOOL bSeparateForm,
    FS\_BOOL bDecodeInlineImg
);
```

**Description**

Creates a new empty page parsing options object.

**Parameter**


---

bTextOnly	[In] Whether only text object is parsed into object list or not.
-----------	--

---

bMarkedContent	[In] Whether load marked content (including foxit tag) information or not.
----------------	--

---

bSeparateForm	[In] Whether Generate FPD_FormObject for form or not.
---------------	---

---

bDecodeInlineImg	[In] Whether decode inline image for better performance. This should be disabled for PDF Editor, to keep the result file size smaller.
------------------	--

---

**Return**

A new empty page parsing options object.

**Head file reference**

fpd\_pageTempl.h: 519

**FPDParseOptionsSetFormGenerateFlag****Syntax**

```
void FPDParseOptionsSetFormGenerateFlag (
    FPD\_ParseOptions opt,
    FS\_BOOL bGernerateForm
);
```

**Description**

Sets whether Generate FPD\_FormObject for form or not.

**Parameter**

---

opt	[In] The input page parsing options object.
-----	---

---

bGernerateForm	[In] Whether Generate FPD_FormObject for form or not.
----------------	---

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 561

**FPDParseOptionsSetInlineImageDecodeFlag****Syntax**

```
void FPDParseOptionsSetInlineImageDecodeFlag (
    FPD\_ParseOptions opt,
    FS\_BOOL bDecodeInlineImg
);
```

**Description**

Sets whether decode inline image for better performance, This should be disabled for PDF Editor, to keep the result file size smaller.

**Parameter**

---

opt	[In] The input page parsing options object.
-----	---

---

bDecodeInlineImg	[In] Whether decode inline image for better performance.
------------------	--

---

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 571

**FPDParseOptionsSetMarkedContentLoadFlag****Syntax**

```
void FPDParseOptionsSetMarkedContentLoadFlag (
    FPD\_ParseOptions opt,
    FS\_BOOL bLoad
);
```

**Description**

Sets whether load marked content (including foxit tag) information or not.

**Parameter**

opt	[In] The input page parsing options object.
bLoad	[In] Whether load marked content (including foxit tag) information or not.

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 551

**FPDParseOptionsSetTextFlag****Syntax**

```
void FPDParseOptionsSetTextFlag (
    FPD\_ParseOptions opt,
    FS\_BOOL bParseTextOnly
);
```

**Description**

Sets whether only text object is parsed into object list or not.

**Parameter**

opt	[In] The input page parsing options object.
bParseTextOnly	[In] Whether only text object is parsed into object list or not.

**Return**

void

**Head file reference**

fpd\_pageTempl.h: 541

## FPD\_Parser

**Return from Used by****Description****Returned from**

[FRDocGetParser](#)

[FPDDocGetParser](#)

[FPDParserNew](#)

**Used by**

[FPDParserCheckEmbeddedSecurity](#)

[FPDParserCheckPassword](#)

[FPDParserCheckStandardSecurityPassword](#)

[FPDParserCloseParser](#)

[FPDParserCountOtherTrailers](#)

[FPDParserDeleteIndirectObject](#)

[FPDParserDestroy](#)

[FPDParserGetDocument](#)

[FPDParserGetEncryptDict](#)

[FPDParserGetFileAccess](#)

[FPDParserGetFileStreamOption](#)

[FPDParserGetFileVersion](#)

[FPDParserGetFirstPageNo](#)

[FPDParserGetIDArray](#)

[FPDParserGetIndirectBinary](#)

[FPDParserGetInfoObjNum](#)

[FPDParserGetLastObjNum](#)

[FPDParserGetLastXRefOffset](#)

[FPDParserGetLengthOfVersions](#)

[FPDParserGetObjectOffset](#)

[FPDParserGetObjectSize](#)

[FPDParserGetObjectVersion](#)

[FPDParserGetOtherTrailerByIndex](#)

[FPDParserGetPassword](#)

[FPDParserGetPermissions](#)

[FPDParserGetRootObjNum](#)

[FPDParserGetSecurityCryptInfo](#)

[FPDParserGetSecurityPermissions](#)

[FPDParserGetStandardSecurityRevision](#)

[FPDParserGetStandardSecurityUserPassword](#)

[FPDParserGetStandardSecurityVersion](#)

[FPDParseGetTrailer](#)  
[FPDParseGetUnicodePassword](#)  
[FPDParseIsFormStream](#)  
[FPDParseIsMetadataEncryptedBySecurity](#)  
[FPDParseIsOwner](#)  
[FPDParseIsSecurityOwner](#)  
[FPDParseIsXRefStream](#)  
[FPDParseLoadAttachmentStream](#)  
[FPDParseParseIndirectObject](#)  
[FPDParseReloadFileStream](#)  
[FPDParseSetFileStreamOption](#)  
[FPDParseSetPassword](#)  
[FPDParseSetUnicodePassword](#)  
[FPDParseStartAsynParse](#)  
[FPDParseStartParse](#)  
[FPDParseStartParseCustomFile](#)  
[FPDParseStartParseFormMem](#)  
[FPDParseStartParseW](#)

## Definitions

### Definitions summary

#### [FPD\\_PARSE\\_NOSTREAM](#)

No stream.

#### [FPD\\_PARSE\\_TYPEONLY](#)

Type only.

#### [FPD\\_PARSE\\_ERROR\\_CERT](#)

This document is encrypted by digital certificate and current user doesn't have correct certificate.

#### [FPD\\_PARSE\\_ERROR\\_FILE](#)

file error: not found or could not be opened.

#### [FPD\\_PARSE\\_ERROR\\_FORMAT](#)

format error: not a PDF or corrupted.

#### [FPD\\_PARSE\\_ERROR\\_HANDLER](#)

This document is encrypted by some unsupported security handler.

#### [FPD\\_PARSE\\_ERROR\\_PASSWORD](#)

Invalid password. Please input again.

#### [FPD\\_PARSE\\_ERROR\\_SUCCESS](#)

no error.

### Definitions detail

#### **FPD\_PARSE\_NOSTREAM**

##### **Syntax**

```
#define FPD_PARSE_NOSTREAM 2
```

##### **Description**

No stream.

##### **Group**

[FPDParseContextFlags](#)

**Head file reference**

fpd\_parserExpT.h: 102

**FPD\_PARSE\_TYPEONLY****Syntax**

#define FPD\_PARSE\_TYPEONLY 1

**Description**

Type only.

**Group**[FPDParseContextFlags](#)**Head file reference**

fpd\_parserExpT.h: 100

**FPD\_PARSE\_ERROR\_CERT****Syntax**

#define FPD\_PARSE\_ERROR\_CERT 5

**Description**

This document is encrypted by digital certificate and current user doesn't have correct certificate.

**Group**[FPDParseErrCodes](#)**Head file reference**

fpd\_parserExpT.h: 87

**FPD\_PARSE\_ERROR\_FILE****Syntax**

#define FPD\_PARSE\_ERROR\_FILE 1

**Description**

file error: not found or could not be opened.

**Group**[FPDParseErrCodes](#)**Head file reference**

fpd\_parserExpT.h: 79

## FPD\_PARSE\_ERROR\_FORMAT

### Syntax

```
#define FPD_PARSE_ERROR_FORMAT 2
```

### Description

format error: not a PDF or corrupted.

### Group

[FPDParseErrCodes](#)

### Head file reference

fpd\_parserExpT.h: 81

## FPD\_PARSE\_ERROR\_HANDLER

### Syntax

```
#define FPD_PARSE_ERROR_HANDLER 4
```

### Description

This document is encrypted by some unsupported security handler.

### Group

[FPDParseErrCodes](#)

### Head file reference

fpd\_parserExpT.h: 85

## FPD\_PARSE\_ERROR\_PASSWORD

### Syntax

```
#define FPD_PARSE_ERROR_PASSWORD 3
```

### Description

Invalid password. Please input again.

### Group

[FPDParseErrCodes](#)

### Head file reference

fpd\_parserExpT.h: 83

## FPD\_PARSE\_ERROR\_SUCCESS

### Syntax

```
#define FPD_PARSE_ERROR_SUCCESS 0
```

**Description**

no error.

**Group**

[FPParseErrCodes](#)

**Head file reference**

fpd\_parserExpT.h: 77

## Structures

### Structures summary

The PDF file parser to parse a pdf file to get a [FPD Document](#).

### Structs detail

**Syntax**

```
typedef struct _t_FPD_Parser* FPD_Parser;
{
    FS_BOOL flags,
    FS_DWORD dictStart,
    FS_DWORD dictEnd,
    FS_DWORD dataStart,
    FS_DWORD dataEnd
};
```

**Description**

The PDF file parser to parse a pdf file to get a [FPD Document](#). Each PDF file need a FPD\_Parser to parse the content, call [FPDParserStartParse](#) to start to parse a file. See [FPDParserNew](#), [FPDParserDestroy](#).

**Head file reference**

fpd\_parserExpT.h: 142

**flags**

The flags for parser context.

**dictStart**

The start position of the dictionary.

**dictEnd**

The end position of the dictionary.

**dataStart**

The start position of the data.

**dataEnd**

The end position of the data.

## Functions

### Functions summary

#### [\*\*FPDParserCheckEmbeddedSecurity\*\*](#)

Checks the access to the encrypted embedded file stream.

#### [\*\*FPDParserCheckPassword\*\*](#)

Check whether the password is verified successfully.

#### [\*\*FPDParserCheckStandardSecurityPassword\*\*](#)

Checks whether the password is verified successfully.

#### [\*\*FPDParserCloseParser\*\*](#)

Closes the parser, as well as the file. If reparsing is used, the document will be kept.

#### [\*\*FPDParserCountOtherTrailers\*\*](#)

Gets the total number of the trailers in a PDF file.

#### [\*\*FPDParserDeleteIndirectObject\*\*](#)

Deletes a indirect object form a file.

#### [\*\*FPDParserDestroy\*\*](#)

Destroy a PDF file parser.

#### [\*\*FPDParserGetDocument\*\*](#)

Gets the [FPD\\_Document](#) object which associate with a PDF file.

#### [\*\*FPDParserGetEncryptDict\*\*](#)

Gets the encrypt dictionary which contain encrypt information.

#### [\*\*FPDParserGetFileAccess\*\*](#)

Gets the stream access interface.

#### [\*\*FPDParserGetFileStreamOption\*\*](#)

Get the file stream option. Loading stream content into memory will improve performance for frequent access, however, it will also consume a lot of memory space. Therefore, we provide an option to leave stream content on file system, and read them whenever we need them. It may reduce the performance a little bit, but greatly reduce the memory consumption, especially when the file is big.

#### [\*\*FPDParserGetFileVersion\*\*](#)

Gets the file version. File version: 14 for 1.4, 15 for 1.5, ...

#### [\*\*FPDParserGetFirstPageNo\*\*](#)

Get the first page number.

#### [\*\*FPDParserGetIDArray\*\*](#)

Gets the ID array in a PDF file.

#### [\*\*FPDParserGetIndirectBinary\*\*](#)

For faster file updating, we can get the binary form of an indirect object, then directly output to destination file. In this case we must call the following function to get the binary buffer.

#### [\*\*FPDParserGetInfoObjNum\*\*](#)

Gets the object number of the file information dictionary.

#### [\*\*FPDParserGetLastObjNum\*\*](#)

Gets the lastest object number.

#### [\*\*FPDParserGetLastXRefOffset\*\*](#)

Gets the offset of last xref.

#### [\*\*FPDParserGetLengthOfVersions\*\*](#)

Gets the version's length of the file.

#### [\*\*FPDParserGetObjectOffset\*\*](#)

Gets offset of indirect object.

#### [\*\*FPDParserGetObjectSize\*\*](#)

Gets data size of indirect object.

**FPDParserGetObjectVersion**

Gets the generation number of indirect object.

**FPDParserGetOtherTrailerByIndex**

Gets the specified trailer dictionary.

**FPDParserGetPassword**

Gets the password string.

**FPDParserGetPermissions**

Gets the use permissions of a PDF file.

**FPDParserGetRootObjNum**

Gets the object number of file catalog dictionary.

**FPDParserGetSecurityCryptInfo**

Gets encryption information including standard algorithm and key.

**FPDParserGetSecurityPermissions**

Gets permission settings of the document.

**FPDParserGetStandardSecurityRevision**

Gets the revision of standard security.

**FPDParserGetStandardSecurityUserPassword**

Gets user password from an owner password.

**FPDParserGetStandardSecurityVersion**

Gets the version of standard security.

**FPDParserGetTrailer**

Gets the trailer dictionary.

**FPDParserGetUnicodePassword**

Get the Unicode-based passwords.

**FPDParserIsFormStream**

Checks if an indirect object is a form stream or not, without actually loading the object.

**FPDParserIsMetadataEncryptedBySecurity**

Checks whether document metadata needs to be encrypted.

**FPDParserIsOwner**

Tests whether the user has the owner permission of the document.

**FPDParserIsSecurityOwner**

Checks whether the current user is owner of the document.

**FPDParserIsXRefStream**

Tests whether cross reference stream is used.

**FPDParserLoadAttachmentStream**

Loads the attachment stream.

**FPDParserNew**

Creates a new PDF file parser.

**FPDParserParseIndirectObject**

Parses the indirect objects.

**FPDParserParseIndirectObjectAt**

Parses an indirect object specified by object position.

**FPDParserParseIndirectObjectsAtRange**

Gets the indirect objects in specify byte range.

**FPDParserParseStreamPos**

Gets position information for a stream:

- Start position (just before "<<") and end position (just after ">>") of the stream's dictionary

- Start position (first byte) and end position (just after the last byte) of the stream's data

**FPDParserReloadFileStream**

Reloads stream content of a specified stream object.

**FPDParserSetFileStreamOption**

Sets the file stream option.

**FPDParserSetPassword**

Sets the password of standard encryption for the parser.

**FPDParserSetUnicodePassword**

Set the unicode password of standard encryption for the parser.

**FPDParserStartAsynParse**

Asynchronous parsing a custom file.

**FPDParserStartParse**

Starts parsing from a file, ANSIC version. Use [FPDParserCloseParse](#) () to end the parsing.

**FPDParserStartParseCustomFile**

Starts parsing a custom file. Use [FPDParserCloseParse](#) () to end the parsing.

**FPDParserStartParseFormMem**

Starts parsing from memory block. Use [FPDParserCloseParse](#) () to end the parsing.

**FPDParserStartParseW**

Starts parsing from a file, Unicode version. Use [FPDParserCloseParse](#) () to end the parsing.

## Functions detail

**FPDParserCheckEmbeddedSecurity****Syntax**

```
FS_DWORD FPDParserCheckEmbeddedSecurity (
    FPD Parser parser,
    FS LPCSTR name
);
```

**Description**

Checks the access to the encrypted embedded file stream.

**Parameter**

parser	[In] The PDF file parser.
name	[In] The name of the crypt filter that should be used by default when encrypting embedded file streams.

**Return**

The returned value can refer to [FPDParseErrCodes](#) .

**Head file reference**

fpd\_parserTempl.h: 529

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1](#)**FPDParserCheckPassword****Syntax**

```
FS_INT32 FPDParserCheckPassword (
    FPD Parser parser,
    FS LPCWSTR password,
    FS DWORD pass_size,
    FS BOOL bOwner,
    FS LPBYTE* key
);
```

**Description**

Check whether the password is verified successfully.

**Parameter**

parser	[In] The PDF file parser.
password	[In] The input password pointer.
pass_size	[In] The size of the password.
bOwner	[In] Whether check the owner password
key	[Out] Pointer to key buffer to receive the encryption key. NULL for don't calculate the key.

**Return**

FALSE for invalid password, TRUE for password verified.

**Head file reference**

fpd\_parserTempl.h: 572

**Note:** Only the PDF2.0 supports for the Unicode-based CheckPassword.

**FPDParserCheckStandardSecurityPassword****Syntax**

```
FS_BOOL FPDParserCheckStandardSecurityPassword (
    FPD Parser parser,
    FS LPCBYTE password,
    FS DWORD pass_size,
    FS BOOL bOwner,
    FS LPBYTE* key
```

---

);

**Description**

Checks whether the password is verified successfully.

**Parameter**

parser	[In] The PDF file parser.
password	[In] The input password pointer.
pass_size	[In] The size of the password.
bOwner	[In] Whether check the owner password.
key	[Out] Pointer to key buffer to receive the encryption key. NULL for don't calculate the key.

**Return**

[FALSE](#) for invalid password, [TRUE](#) for password verified.

**Head file reference**

fpd\_parserTempl.h: 515

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDParserCloseParser****Syntax**

```
void FPDParserCloseParser (
    FPD\_Parser parser,
    FS\_BOOL bReParse
);
```

**Description**

Closes the parser, as well as the file. If reparsing is used, the document will be kept.

**Parameter**

parser	[In] The PDF file parser.
bReParse	[In] A flag indicates whether reparsing is used.

**Return**

void

**Head file reference**

fpd\_parserTempl.h: 86

**FPDParserCountOtherTrailers****Syntax**

```
FS_INT32 FPDParserCountOtherTrailers (
    FPD Parser parser
);
```

**Description**

Gets the total number of the trailers in a PDF file.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The total number of the trailers in a PDF file.

**Head file reference**

fpd\_parserTempl.h: 161

**FPDParserDeleteIndirectObject****Syntax**

```
void FPDParserDeleteIndirectObject (
    FPD Parser parser,
    FS\_DWORD objnum
);
```

**Description**

Deletes a indirect object form a file.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

objnum	[In] The indirect object number to be deleted.
--------	--

---

**Return**

void

**Head file reference**

fpd\_parserTempl.h: 258

**FPDParserDestroy****Syntax**

```
void FPDParserDestroy (
    FPD Parser parser
);
```

**Description**

Destroy a PDF file parser.

**Parameter**

---

parser	[In] The parser to be free.
--------	-----------------------------

---

**Return**

void

**Head file reference**

fpd\_parserTempl.h: 31

**FPDParserGetDocument****Syntax**

```
FPD_Document FPDParserGetDocument (
    FPD Parser parser
);
```

**Description**

Gets the [FPD Document](#) object which associate with a PDF file.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The new empty document.

**Head file reference**

fpd\_parserTempl.h: 23

**FPDParserGetEncryptDict**

**Syntax**

```
FPD_Object FPDParserGetEncryptDict (
    FPD Parser parser
);
```

**Description**

Gets the encrypt dictionary which contain encrypt information.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The encrypt dictionary object.

**Head file reference**

fpd\_parserTempl.h: 207

**FPDParserGetFileAccess****Syntax**

```
FS_FileReadHandler FPDParserGetFileAccess (
    FPD Parser parser
);
```

**Description**

Gets the stream access interface.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

A file access interface handler.

**Head file reference**

fpd\_parserTempl.h: 379

**FPDParserGetFileStreamOption****Syntax**

```
FS_BOOL FPDParserGetFileStreamOption (
    FPD Parser parser
);
```

**Description**

Get the file stream option. Loading stream content into memory will improve performance for frequent access, however, it will also consume a lot of memory space. Therefore, we provide an option to leave stream content on file system, and read them whenever we need them. It may reduce the performance a little bit, but greatly reduce the memory consumption, especially when the file is big.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The new empty document.

**Head file reference**

fpd\_parserTempl.h: 355

**FPDParserGetFileVersion****Syntax**

```
FS_INT32 FPDParserGetFileVersion (
    FPD\_Parser parser
);
```

**Description**

Gets the file version. File version: 14 for 1.4, 15 for 1.5, ...

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The new empty document.

**Head file reference**

fpd\_parserTempl.h: 388

**FPDParserGetFirstPageNo****Syntax**

```
FS_DWORD FPDParserGetFirstPageNo (
    FPD\_Parser parser
);
```

**Description**

Get the first page number.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The index of first page.

**Head file reference**

fpd\_parserTempl.h: 430

**FPDParserGetIDArray****Syntax**

```
FPD_Object FPDParserGetIDArray (
    FPD Parser parser
);
```

**Description**

Gets the ID array in a PDF file.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

A PDF array object which contain the IDs of a PDF file.

**Head file reference**

fpd\_parserTempl.h: 198

**FPDParserGetIndirectBinary****Syntax**

```
void FPDParserGetIndirectBinary (
    FPD Parser parser,
    FS DWORD objnum,
    FS BinaryBuf buffer
);
```

**Description**

For faster file updating, we can get the binary form of an indirect object, then directly output to destination file. In this case we must call the following function to get the binary buffer.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

---

objnum	[In] The indirect object number.
--------	----------------------------------

---

buffer	[In] A binary buffer to receive bytes.
--------	--

---

**Return**

void

**Head file reference**

fpd\_parserTempl.h: 310

**Note:** This function can't be used for any updated PDF file.

**FPDParserGetInfoObjNum****Syntax**

```
FS_DWORD FPDParserGetInfoObjNum (
    FPD Parser parser
);
```

**Description**

Gets the object number of the file information dictionary.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The object number indicates the file information dictionary.

**Head file reference**

fpd\_parserTempl.h: 189

**FPDParserGetLastObjNum****Syntax**

```
FS_DWORD FPDParserGetLastObjNum (
    FPD Parser parser
);
```

**Description**

Gets the lastest object number.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The object number of the last object in a PDF file.

**Head file reference**

fpd\_parserTempl.h: 228

**FPDParserGetLastXRefOffset****Syntax**

```
FS_DWORD FPDParserGetLastXRefOffset (
    FPD Parser parser
);
```

**Description**

Gets the offset of last xref.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The bytes of the xref offset.

**Head file reference**

fpd\_parserTempl.h: 143

**FPDParserGetLengthOfVersions****Syntax**

```
void FPDParserGetLengthOfVersions (
    FPD Parser,
    FS\_DWordArray arrayLengths
);
```

**Description**

Gets the version's length of the file.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

arrayLengths	[In] A DWORD array to receive the length of versions.
--------------	---

---

**Return**

void

**Head file reference**

fpd\_parserTempl.h: 268

**FPDParserGetObjectOffset****Syntax**

```
FS_DWORD FPDParserGetObjectOffset (
    FPD Parser parser,
    FS DWORD objnum
);
```

**Description**

Gets offset of indirect object.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

objnum	[In] The indirect object number.
--------	----------------------------------

---

**Return**

The offset of the indirect object. For objects stored in object stream, this is the offset of the object stream.

**Head file reference**

fpd\_parserTempl.h: 278

**FPDParserGetObjectSize****Syntax**

```
FS_DWORD FPDParserGetObjectSize (
    FPD Parser parser,
    FS DWORD objnum
);
```

**Description**

Gets data size of indirect object.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

objnum	[In] The indirect object number.
--------	----------------------------------

---

**Return**

The data size, in bytes, of the indirect object. For objects stored in object stream, this is the data size of the object stream.

**Head file reference**

fpd\_parserTempl.h: 289

**FPDParserGetObjectVersion****Syntax**

```
int FPDParserGetObjectVersion (
    FPD Parser parser,
    FS DWORD objnum
);
```

**Description**

Gets the generation number of indirect object.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

objnum	[In] The indirect object number.
--------	----------------------------------

---

**Return**

The generation number of the indirect object.

**Head file reference**

fpd\_parserTempl.h: 300

**FPDParserGetOtherTrailerByIndex****Syntax**

```
FPD_Object FPDParserGetOtherTrailerByIndex (
    FPD Parser parser,
    FS INT32 index
);
```

**Description**

Gets the specified trailer dictionary.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

index	[In] The index of trailer which will be obtained, the range is 0 to <a href="#">FPDParserCountOtherTrailers</a> ()-1.
-------	---

---

**Return**

The specified trailer dictionary.

**Head file reference**

fpd\_parserTempl.h: 170

**FPDParserGetPassword****Syntax**

```
void FPDParserGetPassword (
    FPD Parser parser,
    FS ByteString* outPassword
);
```

**Description**

Gets the password string.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

outPassword	[Out] A buffer to receive the password string.
-------------	--

---

**Return**

void

**Head file reference**

fpd\_parserTempl.h: 124

**FPDParserGetPermissions****Syntax**

```
FS_DWORD FPDParserGetPermissions (
    FPD Parser parser
);
```

**Description**

Gets the use permissions of a PDF file.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The permission codes. See [FPDDocPermissions](#).

**Head file reference**

fpd\_parserTempl.h: 96

**FPDParserGetRootObjNum****Syntax**

```
FS_DWORD FPDParserGetRootObjNum (
    FPD Parser parser
);
```

**Description**

Gets the object number of file catalog dictionary.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The object number of file root(catalog) dictionary.

**Head file reference**

fpd\_parserTempl.h: 180

**FPDParserGetSecurityCryptInfo****Syntax**

```
FS_BOOL FPDParserGetSecurityCryptInfo (
    FPD Parser parser,
    FS_INT32* outCipher,
    FS_LPCBYTE* outBuffer,
    FS_INT32* outKeylen
);
```

**Description**

Gets encryption information including standard algorithm and key.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

outCipher	[Out] Receives cipher identifier (FPD_CIPHER_NONE, FPD_CIPHER_RC4 or FPD_CIPHER_AES).
-----------	---

---

outBuffer	[Out] Receives a pointer to the key buffer.
-----------	---

---

---

outKeylen	[Out] Receives number of bytes in the key.
-----------	--

---

**Return**

[TRUE](#) if successful. [FALSE](#) if no standard key info is provided or failure.

**Head file reference**

fpd\_parserTempl.h: 459

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDParserGetSecurityPermissions****Syntax**

```
FS_DWORD FPDParserGetSecurityPermissions (
    FPD Parser parser
);
```

**Description**

Gets permission settings of the document.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The permission settings of the document.

**Head file reference**

fpd\_parserTempl.h: 439

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDParserGetStandardSecurityRevision****Syntax**

```
FS_INT32 FPDParserGetStandardSecurityRevision (
    FPD Parser parser
);
```

**Description**

Gets the revision of standard security.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The revision of standard security.

**Head file reference**

fpd\_parserTempl.h: 505

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FPDParserGetStandardSecurityUserPassword****Syntax**

```
void FPDParserGetStandardSecurityUserPassword (
    FPD\_Parser parser,
    FS\_LPCBYTE owner_pass,
    FS\_DWORD pass_size,
    FS\_BytString* outUserPsswrd
);
```

**Description**

Gets user password from an owner password.

**Parameter**


---

parser	[In] The PDF file parser.
--------	---------------------------

---

owner_pass	[In] The owner password pointer.
------------	----------------------------------

---

pass_size	[In] The length of the owner password.
-----------	--

---

outUserPsswrd	[Out] Receives number of bytes in the key.
---------------	--

---

**Return**

It receives the correspond user password for this standard security handler.

**Head file reference**

fpd\_parserTempl.h: 482

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FPDParserGetStandardSecurityVersion**

**Syntax**

```
FS_INT32 FPDParserGetStandardSecurityVersion (
    FPD Parser parser
);
```

**Description**

Gets the version of standard security.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The version of standard security.

**Head file reference**

fpd\_parserTempl.h: 495

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDParserGetTrailer****Syntax**

```
FPD_Object FPDParserGetTrailer (
    FPD Parser parser
);
```

**Description**

Gets the trailer dictionary.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

The trailer dictionary.

**Head file reference**

fpd\_parserTempl.h: 134

**FPDParserGetUnicodePassword****Syntax**

```
void FPDParserGetUnicodePassword (
    FPD Parser parser,
    FS\_WideString* outPassword
```

---

);

**Description**

Get the Unicode-based passwords.

**Parameter**


---

parser	[In] The PDF file parser.
outPassword	[Out] A buffer to receive the Unicode-based password string.

---

**Return**

void

**Head file reference**

fpd\_parserTempl.h: 562

**Note:** Only the PDF2.0 supports for the Unicode-based passwords.

**FPDParserIsFormStream****Syntax**

```
FS_BOOL FPDParserIsFormStream (
    FPD\_Parser parser,
    FS\_DWORD objnum,
    FS\_BOOL* bForm
);
```

**Description**

Checks if an indirect object is a form stream or not, without actually loading the object.

**Parameter**


---

parser	[In] The PDF file parser.
objnum	[In] The stream object number.
bForm	[Out] It receive whether it's a form stream or not.

---

**Return**

Non-zero means determined, otherwise unknown.

**Head file reference**

fpd\_parserTempl.h: 237

## FPDParserIsMetadataEncryptedBySecurity

### Syntax

```
FS_BOOL FPDParserIsMetadataEncryptedBySecurity (
    FPD Parser parser
);
```

### Description

Checks whether document metadata needs to be encrypted.

### Parameter

---

parser	[In] The PDF file parser.
--------	---------------------------

---

### Return

Whether document metadata needs to be encrypted.

### Head file reference

fpd\_parserTempl.h: 472

### Since

[SDK LATEEST VERSION > 1.0](#)

## FPDParserIsOwner

### Syntax

```
FS_BOOL FPDParserIsOwner (
    FPD Parser parser
);
```

### Description

Tests whether the user has the owner permission of the document.

### Parameter

---

parser	[In] The PDF file parser.
--------	---------------------------

---

### Return

[TRUE](#) for owner permission, otherwise [FALSE](#).

### Head file reference

fpd\_parserTempl.h: 105

## FPDParserIsSecurityOwner

### Syntax

```
FS_BOOL FPDParserIsSecurityOwner (
```

```
FPD_Parser parser  
);
```

**Description**

Checks whether the current user is owner of the document.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

Whether the current user is owner of the document.

**Head file reference**

fpd\_parserTempl.h: 449

**Since**

[SDK LATEEST VERSION > 1.0](#)

**FPDParserIsXRefStream****Syntax**

```
FS_BOOL FPDParserIsXRefStream (  
    FPD_Parser parser  
);
```

**Description**

Tests whether cross reference stream is used.

**Parameter**

---

parser	[In] The PDF file parser.
--------	---------------------------

---

**Return**

[TRUE](#) for xref stream, otherwise [FALSE](#).

**Head file reference**

fpd\_parserTempl.h: 397

**FPDParserLoadAttachmentStream****Syntax**

```
FS_DWORD FPDParserLoadAttachmentStream (  
    FPD_Parser parser,  
    FS_DWORD objnum,  
    FPD_Object* outStremObj  
);
```

**Description**

Loads the attachment stream.

**Parameter**


---

parser	[In] The PDF file parser.
objnum	[In] The object number of the attachment stream.
outStremObj	[Out] It receives the pointer to the loaded attachment stream object.

---

**Return**

The returned value can refer to [FPD Parse Err Codes](#).

**Head file reference**

fpd\_parserTempl.h: 540

**Since**

[SDK LATEST VERSION > 2.1](#)

**FPDParserNew****Syntax**

```
FPD_Parser FPDParserNew (void );
```

**Description**

Creates a new PDF file parser. The PDF file parser will parse entire PDF file and then build a [FPD Document](#) object, use [FPDParserGetDocument\(\)](#) to get a [FPD Document](#) object.

**Return**

The newly created parser.

**Head file reference**

fpd\_parserTempl.h: 21

**FPDParserParseIndirectObject****Syntax**

```
FPD_Object FPDParserParseIndirectObject (
    FPD Parser parser,
    FPD Document objList,
    FS DWORD objnum,
    FPD PARSE CONTEXT\* pContext
);
```

**Description**

Parses the indirect objects.

**Parameter**

parser	[In] The PDF file parser.
objList	[In] The <a href="#">FPD Document</a> object which contains all indirect objects.
objnum	[In] The object number.
pContext	[In] The parse context. See <a href="#">FPD_PARSE_CONTEXT</a> .

**Return**

A PDF object.

**Head file reference**

fpd\_parserTempl.h: 216

**FPDParserParseIndirectObjectAt****Syntax**

```
FPD_Object FPDParserParseIndirectObjectAt (
    parser,
    FPD Document objList,
    FS\_DWORD pos,
    FS\_DWORD objnum,
    FPD\_PARSE\_CONTEXT* pContext
);
```

**Description**

Parses an indirect object specified by object position.

**Parameter**

parser	[In] The PDF file parser.
objList	[In] The <a href="#">FPD Document</a> object that contain the indirect objects.
pos	[In] The position specified the indirect object.

---

objnum	[In] The indirect object number.
--------	----------------------------------

---

pContext	[In] The parse context.
----------	-------------------------

---

**Return**

A PDF object.

**Head file reference**

fpd\_parserTempl.h: 406

**FPDParserParseIndirectObjectsAtRange****Syntax**

```
FS_BOOL FPDParserParseIndirectObjectsAtRange (
    parser,
    FS_DWordArray objNum,
    FS_DWordArray objPos,
    FS_DWORD dwPos,
    FS_DWORD dwLen,
    FPD_PARSE_CONTEXT* context
);
```

**Description**

Gets the indirect objects in specify byte range.

**Parameter**


---

parser	[In] The PDF file parser.
--------	---------------------------

---

objNum	[In] The indirect object number.
--------	----------------------------------

---

objPos	[In] The indirect object's start position.
--------	--

---

dwPos	[In] The start position.
-------	--------------------------

---

dwLen	[In] The length will be parsed.
-------	---------------------------------

---

context	[In] The parse context.
---------	-------------------------

---

**Return**

Non\_zero means parse successfully, otherwise not.

**Head file reference**

fpd\_parserTempl.h: 341

**Note:**This function is bitwise search so very slow.

### FPDParserParseStreamPos

#### Syntax

```
FS_BOOL FPDParserParseStreamPos (
    parser,
    FS_DWORD objnum,
    FS_DWORD* dict_start,
    FS_DWORD* dict_end,
    FS_DWORD* data_start,
    FS_DWORD* data_end
);
```

#### Description

Gets position information for a stream:

- Start position (just before "<<") and end position (just after ">") of the stream's dictionary
- Start position (first byte) and end position (just after the last byte) of the stream's data

#### Parameter

parser	[In] The PDF file parser.
objnum	[In] The indirect object number.
dict_start	[Out] Start position of the stream's dictionary.
dict_end	[Out] End position of the stream's dictionary.
data_start	[Out] Start position of the stream's data.
data_end	[Out] End position of the stream's data.

#### Return

Non-zero means parse successfully, otherwise not.

#### Head file reference

fpd\_parserTempl.h: 323

### FPDParserReloadFileStream

**Syntax**

```
FS_BOOL FPDParserReloadFileStream (
    FPD Parser parser,
    FPD Object stream
);
```

**Description**

Reloads stream content of a specified stream object.

**Parameter**

---

parser	[In] The PDF file parser.
stream	[In] The stream object which content stream will be reload.

---

**Return**

[TRUE](#) for success, otherwise [FALSE](#).

**Head file reference**

fpd\_parserTempl.h: 248

**FPDParserSetFileStreamOption****Syntax**

```
void FPDParserSetFileStreamOption (
    FPD Parser parser,
    FS BOOL b
);
```

**Description**

Sets the file stream option.

**Parameter**

---

parser	[In] The PDF file parser.
b	[In] A flag indicates whether the stream is a file stream.

---

**Return**

void

**Head file reference**

fpd\_parserTempl.h: 369

**FPDParserSetPassword**

**Syntax**

```
void FPDParserSetPassword (
    FPD Parser parser,
    const FS\_CHAR\* password
);
```

**Description**

Sets the password of standard encryption for the parser.

**Parameter**

parser	[In] The PDF file parser.
password	[In] The input password string.

**Return**

void

**Head file reference**

fpd\_parserTempl.h: 114

**FPDParserSetUnicodePassword****Syntax**

```
FS_BOOL FPDParserSetUnicodePassword (
    FPD Parser parser,
    const FS\_WCHAR\* password
);
```

**Description**

Set the unicode password of standard encryption for the parser.

**Parameter**

parser	[In] The PDF file parser.
password	[In] The input unicode password string.

**Return**

Zero means the unicode password strings include invalid characters.

**Head file reference**

fpd\_parserTempl.h: 552

**Note:** Only the PDF2.0 supports for the Unicode-based passwords.

**FPDParserStartAsynParse****Syntax**

```
FS_DWORD FPDParserStartAsynParse (
    FPD Parser parser,
    FS\_FileReadHandler pFile,
    FS\_BOOL bReParse
);
```

**Description**

Asynchronous parsing a custom file.

**Parameter**


---

parser	[In] The PDF file parser.
--------	---------------------------

---

pFile	[In] The stream access interface handler.
-------	---

---

bReParse	[In] A flag indicates whether we will do reparsing.
----------	---

---

**Return**

The status of PDF parsing.

**Head file reference**

[fpd\\_parserTempl.h](#): 419

**FPDParserStartParse****Syntax**

```
FS_DWORD FPDParserStartParse (
    FPD Parser parser,
    FS\_LPCSTR filename,
    FS\_BOOL bReParse
);
```

**Description**

Starts parsing from a file, ANSIC version. Use [FPDParserCloseParse](#) () to end the parsing.

**Parameter**


---

parser	[In] The PDF file parser.
--------	---------------------------

---

filename	[In] The file full path name that will be parsed.
----------	---

---

bReParse	[In] A flag indicates whether you will do reparsing.
----------	--

---

**Return**

The status of PDF parsing. See [FPDParserErrCodes](#) .

**Head file reference**

fpd\_parserTempl.h: 40

**FPDParserStartParseCustomFile****Syntax**

```
FS_DWORD FPDParserStartParseCustomFile (
    FPD\_Parser parser,
    FS\_FileReadHandler file,
    FS\_BOOL bReParse,
    FS\_BOOL bOwnFileRead
);
```

**Description**

Starts parsing a custom file. Use [FPDParserCloseParse](#) () to end the parsing.

**Parameter**

parser	[In] The PDF file parser.
file	[In] The stream access interface. Use <a href="#">FSFileReadHandlerNew</a> () to create a file access interface.
bReParse	[In] A flag indicates whether you will do reparsing.
bOwnFileRead	[In] A flag indicated whether <a href="#">FPD_Parser</a> takes ownership of the file read structure (by calling <a href="#">FSFileReadHandlerDestroy</a> () when parser cleans up).

**Return**

The status of PDF parsing. See [FPDParserErrCodes](#)

**Head file reference**

fpd\_parserTempl.h: 74

**FPDParserStartParseFormMem****Syntax**

```
FS_DWORD FPDParserStartParseFormMem (
    FPD\_Parser parser,
    FS\_LPCBYTE pData,
    FS\_DWORD nSize,
    FS\_BOOL bReParse
);
```

**Description**

Starts parsing from memory block. Use [FPDParserCloseParse](#) () to end the parsing.

**Parameter**

parser	[In] The PDF file parser.
pData	[In] The input memory block that contains the PDF file data.
nSize	[In] The size in bytes of the memory block.
bReParse	[In] A flag indicates whether you will do reparsing.

**Return**

The status of PDF parsing. See [FPDParseErrCodes](#)

**Head file reference**

fpd\_parserTempl.h: 62

**FPDParserStartParseW****Syntax**

```
FS_DWORD FPDParserStartParseW (
    FPD Parser parser,
    FS\_LPCWSTR filename,
    FS\_BOOL bReParse
);
```

**Description**

Starts parsing from a file, Unicode version. Use [FPDParserCloseParse](#) () to end the parsing.

**Parameter**

parser	[In] The PDF file parser.
filename	[In] The file full path name that will be parsed.
bReParse	[In] A flag indicates whether you will do reparsing.

**Return**

The status of PDF parsing. See [FPDParseErrCodes](#)

**Head file reference**  
fpd\_parserTempl.h: 51

## FPD\_Path

### [Return from Used by](#)

### Description

FPD path. A [FPD\\_Path](#) is a data container for a [FPD\\_PathObject](#) which is a graphic object representing a path in a page description. See [FPDPathNew](#), [FPDPathDestroy](#).

### Returned from

[FPDClipPathGetPathPointer](#)  
[FPDFontLoadGlyphPath](#)  
[FPDPathObjectGetPath](#)  
[FPDPathNew](#)

### Used by

[FPDClipPathAppendPath](#)  
[FPDClipPathGetPath](#)  
[FPDPageObjectAppendClipPath](#)  
[FPDRenderDeviceDrawPath](#)  
[FPDRenderDeviceDrawTextPath](#)  
[FPDRenderDeviceSetClip\\_PathFill](#)  
[FPDPathAppend](#)  
[FPDPathAppendRect](#)  
[FPDPathDestroy](#)  
[FPDPathGetBoundingBox](#)  
[FPDPathGetBoundingBox2](#)  
[FPDPathGetFlag](#)  
[FPDPathGetModify](#)  
[FPDPathGetPoint](#)  
[FPDPathGetPointCount](#)  
[FPDPathGetPointX](#)  
[FPDPathGetPointY](#)  
[FPDPathIsRect](#)  
[FPDPathSetPoint](#)  
[FPDPathSetPointCount](#)  
[FPDPathTransform](#)

## Functions

### Functions summary

#### [FPDPathAppend](#)

Appends a path. Optionally a matrix can be specified to transform the source path before appending.

#### [FPDPathAppendRect](#)

Appends a rectangle.

**FPDPathDestroy**

Destroys the PDF path data object.

**FPDPathGetBoundingBox**

Gets bounding box of all control points.

**FPDPathGetBoundingBox2**

Calculates bounding box (guaranteed to contain all path, may be larger) for stroked path.

**FPDPathGetFlag**

Gets the flag of specified path point.

**FPDPathGetModify**

The interface helps init the object if the object is NULL.

**FPDPathGetPoint**

Gets the path point in the path points array by the index.

**FPDPathGetPointCount**

Gets the point count int the path.

**FPDPathGetPointX**

Gets the x-coordinate of specified path point.

**FPDPathGetPointY**

Gets the y-coordinate of specified path point.

**FPDPathIsRect**

Tests whether the path is a actually rectangle.

**FPDPathNew**

Creates a new empty PDF path data object.

**FPDPathSetPoint**

Sets the point data for specified path point.

**FPDPathSetPointCount**

Changes the path point count and prepares adequate allocated buffer.

**FPDPathTransform**

Transforms this path.

## Functions detail

### **FPDPathAppend**

#### **Syntax**

```
void FPDPathAppend (
    FPD\_Path path,
    FPD\_Path src,
    FS\_AffineMatrix* pMatrix
);
```

#### **Description**

Appends a path. Optionally a matrix can be specified to transform the source path before appending.

#### **Parameter**

---

path	[In] The input PDF path data object.
------	--------------------------------------

---

---

src	[In] The source path.
-----	-----------------------

---

pMatrix	[In] The specified matrix. <a href="#">NULL</a> means no transformation.
---------	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 118

**FPDPathAppendRect****Syntax**

```
void FPDPathAppendRect (
    FPD\_Path path,
    FS\_FLOAT left,
    FS\_FLOAT bottom,
    FS\_FLOAT right,
    FS\_FLOAT top
);
```

**Description**

Appends a rectangle.

**Parameter**


---

path	[In] The input PDF path data object.
------	--------------------------------------

---

left	[In] The x-coordinate of the left-bottom corner.
------	--

---

bottom	[In] The y-coordinate of the left-bottom corner.
--------	--

---

right	[In] The x-coordinate of the right-top corner.
-------	--

---

top	[In] The y-coordinate of the right-top corner.
-----	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 129

**FPDPathDestroy****Syntax**

```
void FPDPathDestroy (
    FPD\_Path path
);
```

**Description**

Destroys the PDF path data object.

**Parameter**

---

path	[In] The input PDF path data object.
------	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 29

**FPDPathGetBoundingBox****Syntax**

```
FS_FloatRect FPDPathGetBoundingBox (
    FPD\_Path path
);
```

**Description**

Gets bounding box of all control points.

**Parameter**

---

path	[In] The input PDF path data object.
------	--------------------------------------

---

**Return**

The bounding box of all control points.

**Head file reference**

fpd\_pageobjTempl.h: 87

**Note:** The result can be used as the bounding box of the whole filled path. However, when path is stroked using geometry pen, the actual bounding box can be much larger.

**FPDPathGetBoundingBox2****Syntax**

```
FS_FloatRect FPDPathGetBoundingBox2 (
    FPD\_Path path,
    FS\_FLOAT lineWidth,
    FS\_FLOAT miterLimit
```

);

**Description**

Calculates bounding box (guaranteed to contain all path, may be larger) for stroked path.

**Parameter**

path	[In] The input PDF path data object.
lineWidth	[In] The line width used in stroking.
miterLimit	[In] The miter limit value for line joint in stroking.

**Return**

The bounding box for stroked path.

**Head file reference**

fpd\_pageobjTempl.h: 97

**FPDPathGetFlag****Syntax**

```
FS_INT32 FPDPathGetFlag (
    FPD\_Path path,
    FS_INT32 index
);
```

**Description**

Gets the flag of specified path point.

**Parameter**

path	[In] The input PDF path data object.
index	[In] Specifies the zero-based index of path point in the path.

**Return**

The flag of specified path point.

**Head file reference**

fpd\_pageobjTempl.h: 47

**FPDPathGetModify**

**Syntax**

```
void FPDPathGetModify (
    FPD\_Path path
);
```

**Description**

The interface helps init the object if the object is NULL.

**Parameter**

---

path	[In] The input PDF path data object.
------	--------------------------------------

---

**Return**

void.

**Head file reference**

[fpd\\_pageobjTempl.h: 174](#)

**Since**

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

**FPDPathGetPoint****Syntax**

```
FS_PATHPOINT FPDPathGetPoint (
    FPD\_Path path,
    FS\_INT32 index
);
```

**Description**

Gets the path point in the path points array by the index.

**Parameter**

---

path	[In] The input PDF path data object.
------	--------------------------------------

---

---

index	[In] The point index in the points array.
-------	---

---

**Return**

The path point in the path points array by the index.

**Head file reference**

[fpd\\_pageobjTempl.h: 38](#)

**FPDPathGetPointCount**

**Syntax**

```
FS_INT32 FPDPathGetPointCount (
    FPD Path path
);
```

**Description**

Gets the point count int the path.

**Parameter**

---

path	[In] The input PDF path data object.
------	--------------------------------------

---

**Return**

The point count int the path.

**Head file reference**

fpd\_pageobjTempl.h: 38

**FPDPathGetPointX****Syntax**

```
FS_FLOAT FPDPathGetPointX (
    FPD Path path,
    FS\_INT32 index
);
```

**Description**

Gets the x-coordinate of specified path point.

**Parameter**

---

path	[In] The input PDF path data object.
------	--------------------------------------

---

index	[In] Specifies the zero-based index of path point in the path.
-------	--

---

**Return**

The x-coordinate of specified path point.

**Head file reference**

fpd\_pageobjTempl.h: 57

**FPDPathGetPointY****Syntax**

```
FS_FLOAT FPDPathGetPointY (
    FPD Path path,
    FS\_INT32 index
);
```

**Description**

Gets the y-coordinate of specified path point.

**Parameter**

---

path	[In] The input PDF path data object.
------	--------------------------------------

---

index	[In] Specifies the zero-based index of path point in the path.
-------	--

---

**Return**

The y-coordinate of specified path point.

**Head file reference**

fpd\_pageobjTempl.h: 67

**FPDPathIsRect****Syntax**

```
FS_BOOL FPDPathIsRect (
    FPD\_Path path
);
```

**Description**

Tests whether the path is a actually rectangle.

**Parameter**

---

path	[In] The input PDF path data object.
------	--------------------------------------

---

**Return**

[TRUE](#) if the path is actually a rectangle, otherwise not.

**Head file reference**

fpd\_pageobjTempl.h: 142

**FPDPathNew****Syntax**

```
FPD_Path FPDPathNew (void );
```

**Description**

Creates a new empty PDF path data object.

**Return**

A new empty PDF path data object.

**Head file reference**

fpd\_pageobjTempl.h: 21

**FPDPathSetPoint****Syntax**

```
void FPDPathSetPoint (
    FPD\_Path path,
    FS\_INT32 index,
    FS\_FLOAT x,
    FS\_FLOAT y,
    FS\_INT32 flag
);
```

**Description**

Sets the point data for specified path point.

**Parameter**

---

path	[In] The input PDF path data object.
index	[In] Specifies the zero-based index of path point in the path.
x	[In] The x-coordinate of the point to set.
y	[In] The y-coordinate of the point to set.
flag	[In] The flag of the point to set.

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 151

**FPDPathSetPointCount****Syntax**

```
void FPDPathSetPointCount (
    FPD\_Path path,
    FS\_INT32 nPoints
);
```

**Description**

Changes the path point count and prepares adequate allocated buffer.

**Parameter**

path	[In] The input PDF path data object.
nPoints	[In] The new count of path point to change.

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 151

**FPDPathTransform****Syntax**

```
void FPDPathTransform (
    FPD\_Path path,
    FS\_AffineMatrix matrix
);
```

**Description**

Transforms this path.

**Parameter**

path	[In] The input PDF path data object.
matrix	[In] The input matrix used to transform.

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 108

# FPD\_PathObject

## Description

A FPD\_PathObject is a graphic object (a subclass of FPD\_PageObject) representing a path in a page description. A path object is an arbitrary shape made up of straight lines, rectangles, and cubic Bézier curves. A path may intersect itself and may have disconnected sections and holes. A path object ends with one or more painting operators that specify whether the path is stroked, filled, used as a clipping boundary, or some combination of these operations.

## Functions

### Functions summary

#### [FPDPathObjectCalcBoundingBox](#)

Calculates the bounding box.

#### [FPDPathObjectDestroy](#)

Destroys the PDF path object.

#### [FPDPathObjectGetFillMode](#)

Gets the filling mode of the page object.

#### [FPDPathObjectGetPath](#)

Gets the path data object. Reference to path data.

#### [FPDPathObjectGetTransformMatrix](#)

Gets the transformation matrix.

#### [FPDPathObjectIsStrokeMode](#)

Tests whether the paint mode for the path object is stroking mode.

#### [FPDPathObjectNew](#)

Creates a new empty PDF path object.

#### [FPDPathObjectSetFillMode](#)

Sets the new filling mode.

#### [FPDPathObjectSetGraphState](#)

Sets the graph state.

#### [FPDPathObjectSetStrokeMode](#)

Sets whether to stroke the path.

#### [FPDPathObjectSetTransformMatrix](#)

Sets the transformation matrix.

#### [FPDPathObjectTransform](#)

Transforms the path object.

### Functions detail

#### FPDPathObjectCalcBoundingBox

##### Syntax

```
void FPDPathObjectCalcBoundingBox (
    FPD\_PageObject objPath
);
```

##### Description

Calculates the bounding box.

##### Parameter

---

objPath	[In] The input PDF path object.
---------	---------------------------------

---

##### Return

void

##### Head file reference

fpd\_pageobjTempl.h: 1529

## FPDPathObjectDestroy

### Syntax

```
void FPDPathObjectDestroy (
    FPD\_PageObject objPath
);
```

### Description

Destroys the PDF path object.

### Parameter

---

objPath	[In] The input PDF path object.
---------	---------------------------------

---

### Return

void

### Head file reference

fpd\_pageobjTempl.h: 1500

## FPDPathObjectGetFillMode

### Syntax

```
FS_INT32 FPDPathObjectGetFillMode (
    FPD\_PageObject objPath
);
```

### Description

Gets the filling mode of the page object.

### Parameter

---

objPath	[In] The input PDF path object.
---------	---------------------------------

---

### Return

The filling mode code. See [FSFillingModeFlags](#).

### Head file reference

fpd\_pageobjTempl.h: 1586

## FPDPathObjectGetPath

### Syntax

```
FPD_Path FPDPathObjectGetPath (
    FPD\_PageObject objPath
);
```

);

**Description**

Gets the path data object. Reference to path data.

**Parameter**

---

objPath	[In] The input PDF path object.
---------	---------------------------------

---

**Return**

The path data object.

**Head file reference**

fpd\_pageobjTempl.h: 1558

**FPDPathObjectGetTransformMatrix****Syntax**

```
void FPDPathObjectGetTransformMatrix (
    FPD\_PageObject objPath,
    FS\_AffineMatrix* outmatrix
);
```

**Description**

Gets the transformation matrix.

**Parameter**

---

objPath	[In] The input PDF path object.
---------	---------------------------------

---

outmatrix	[Out] It receives the transformation matrix.
-----------	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1538

**Note:** Transformation matrix used to transform the path coordinates. Also used to determine line geometry.

**FPDPathObjectIsStrokeMode****Syntax**

```
FS_BOOL FPDPathObjectIsStrokeMode (
    FPD\_PageObject objPath
);
```

**Description**

Tests whether the paint mode for the path object is stroking mode.

**Parameter**

---

objPath	[In] The input PDF path object.
---------	---------------------------------

---

**Return**

[TRUE](#) to stroke the path, otherwise not.

**Head file reference**

fpd\_pageobjTempl.h: 1567

**FPDPathObjectNew****Syntax**

```
FPD_PageObject FPDPathObjectNew (void );
```

**Description**

Creates a new empty PDF path object.

**Return**

A new empty PDF path object.

**Head file reference**

fpd\_pageobjTempl.h: 1491

**FPDPathObjectSetFillMode****Syntax**

```
void FPDPathObjectSetFillMode (
    FPD\_PageObject objPath,
    FS\_INT32 mode
);
```

**Description**

Sets the new filling mode.

**Parameter**

---

objPath	[In] The input PDF path object.
---------	---------------------------------

---

---

mode	[In] The new filling mode. See <a href="#">FSFillModeFlags</a> .
------	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1595

**FPDPathObjectSetGraphState****Syntax**

```
void FPDPathObjectSetGraphState (
    FPD\_PageObject objPath,
    FPD\_GraphState graphState
);
```

**Description**

Sets the graph state.

**Parameter**

---

objPath	[In] The input PDF path object.
---------	---------------------------------

---

graphState	[In] The input new graph state.
------------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1519

**FPDPathObjectSetStrokeMode****Syntax**

```
void FPDPathObjectSetStrokeMode (
    FPD\_PageObject objPath,
    FS\_BOOL bStroke
);
```

**Description**

Sets whether to stroke the path.

**Parameter**

---

objPath	[In] The input PDF path object.
---------	---------------------------------

---

bStroke	[In] True to stroke the path, otherwise not.
---------	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1576

**FPDPathObjectSetTransformMatrix****Syntax**

```
void FPDPathObjectSetTransformMatrix (
    FPD\_PageObject objPath,
    const FS\_AffineMatrix* matrix
);
```

**Description**

Sets the transformation matrix.

**Parameter**

---

objPath	[In] The input PDF path object.
---------	---------------------------------

---

matrix	[In] The transformation matrix.
--------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1548

**Note:** Transformation matrix used to transform the path coordinates. Also used to determine line geometry.

**FPDPathObjectTransform****Syntax**

```
void FPDPathObjectTransform (
    FPD\_PageObject objPath,
    FS\_AffineMatrix matrix
);
```

**Description**

Transforms the path object.

**Parameter**

---

objPath	[In] The input PDF path object.
---------	---------------------------------

---

matrix	[In] The transformation matrix.
--------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1509

## FPD\_Pattern

**[Return from Used by](#)**

### Description

The abstract class for tiling pattern and shading pattern. Has no "new" functions. see *CHAPTER 4.6 in PDF reference* . See [FPDPatternDestroy](#) .

### Returned from

[FPDCColorGetPattern](#)[FPDDocLoadPattern](#)

### Used by

[FPDCColorSetValue2](#)[FPDCColorStateSetFillPatternColor](#)[FPDCColorStateSetStrokePatternColor](#)[FPDPatternDestroy](#)[FPDPatternGetPatternMatrix](#)[FPDPatternGetPatternObj](#)[FPDPatternGetPatternType](#)[FPDPatternGetPDDoc](#)

## Functions

### Functions summary

**[FPDPatternDestroy](#)**

Destroys the PDF pattern.

**[FPDPatternGetPatternMatrix](#)**

Gets matrix from pattern to parent stream.

**[FPDPatternGetPatternObj](#)**

Gets dictionary for shading, stream for tiling.

**[FPDPatternGetPatternType](#)**

Gets the pattern type

**[FPDPatternGetPDDoc](#)**

Gets the PDF document.

## Functions detail

### FPDPatternDestroy

#### Syntax

```
void FPDPatternDestroy (
    FPD Pattern pattern
);
```

#### Description

Destroys the PDF pattern.

#### Parameter

---

pattern	[In] The input PDF pattern.
---------	-----------------------------

---

#### Return

void

#### Head file reference

fpd\_resourceTempl.h: 1573

**Note:** Can't construct a FPD\_Pattern directly.

### FPDPatternGetPatternMatrix

#### Syntax

```
FS_AffineMatrix FPDPatternGetPatternMatrix (
    FPD Pattern pattern
);
```

#### Description

Gets matrix from pattern to parent stream.

#### Parameter

---

pattern	[In] The input PDF pattern.
---------	-----------------------------

---

#### Return

The matrix from pattern to parent stream.

#### Head file reference

fpd\_resourceTempl.h: 1600

### FPDPatternGetPatternObj

**Syntax**

```
FPD_Object FPDPatternGetPatternObj (
    FPD Pattern pattern
);
```

**Description**

Gets dictionary for shading, stream for tiling.

**Parameter**

---

pattern	[In] The input PDF pattern.
---------	-----------------------------

---

**Return**

A dictionary for shading, stream for tiling.

**Head file reference**

fpd\_resourceTempl.h: 1582

**FPDPatternGetPatternType****Syntax**

```
FS_INT32 FPDPatternGetPatternType (
    FPD Pattern pattern
);
```

**Description**

Gets the pattern type

**Parameter**

---

pattern	[In] The input PDF pattern.
---------	-----------------------------

---

**Return**

The pattern type

**Head file reference**

fpd\_resourceTempl.h: 1591

**FPDPatternGetPDDoc****Syntax**

```
FPD_Document FPDPatternGetPDDoc (
    FPD Pattern pattern
);
```

**Description**

Gets the PDF document.

#### Parameter

pattern	[In] The input PDF pattern.
---------	-----------------------------

#### Return

The PDF document that is the pattern associated with.

#### Head file reference

fpd\_resourceTempl.h: 1609

## FPD\_ProgressiveEncryptHandler

### [Return from Used by](#)

#### Description

The [FPD\\_ProgressiveEncryptHandler](#) object is used to encrypt the file progressively. It is returned by [FPDCreatorSetProgressiveEncryptHandler](#) and can be released by [FPDCreatorReleaseProgressiveEncryptHandler](#).

#### Returned from

[FPDCreatorSetDRMProgressiveEncryptHandler](#)  
[FPDCreatorSetProgressiveEncryptHandler](#)

#### Used by

[FPDCreatorReleaseDRMProgressiveEncryptHandler](#)  
[FPDCreatorReleaseProgressiveEncryptHandler](#)

#### Callbacks

##### Callbacks summary

[FPDProgressiveEncrypGetTempFile](#)  
A callback for progressive encrypt handler.  
[FPDProgressiveEncrypReleaseTempFile](#)  
A callback for progressive encrypt handler.

##### Callbacks detail

###### FPDProgressiveEncrypGetTempFile

###### Syntax

```
typedef FS_FileStream (*FPDProgressiveEncrypGetTempFile)(
    FS_LVOID clientData
);
```

**Description**

A callback for progressive encrypt handler. It is called when the PDF creator is encrypting content progressively. If the stream is very large, the plug-in can provide a temporary local file to store the data, or the creating may fail because of lacking of system memory. Foxit Reader will use memory if the plug-in returns null.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

**Return**

The [FS\\_FileStream](#) object used to store and read the temporary data.

**Head file reference**

fpd\_serialExpT.h: 182

**Group**

[FPD\\_ProgressiveEncryptCallbacksRec](#)

**Related method**

[FPDCreatorSetProgressiveEncryptHandler](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.3](#)

## FPDProgressiveEncrypReleaseTempFile

**Syntax**

```
typedef void (*FPDProgressiveEncrypReleaseTempFile)(  
    FS\_LVOID clientData,  
    FS\_FileStream fileStream  
)
```

**Description**

A callback for progressive encrypt handler. It is called when the PDF creator completes encrypting content progressively. Then the plug-in can release the [FS\\_FileStream](#) object.

**Parameter**

---

clientData	[In] The user-supplied data.
------------	------------------------------

---

fileStream	[In] The <a href="#">FS_FileStream</a> object to be released.
------------	---

---

**Return**

void.

**Head file reference**

fpd\_serialExpT.h: 198

**Group**

[FPD\\_ProgressiveEncryptCallbacksRec](#)

**Related method**

[FPDCreatorSetProgressiveEncryptHandler](#)

**Since**

[SDK LATEEST VERSION > 2.1.0.3](#)

## FPD\_ProgressiveRender

[Return from Used by](#)

### Description

The PDF progressive renderer. A progressive renderer that breaks the full rendering process into steps. This class must be overloaded in order to pause a step whenever the application thinks necessary. Application must first call Start() to start the rendering, then, when it's paused, the application ust call Continue() to continue the rendering, until status becomes Done or Failed. To stop the rendering, simply destruct the renderer. See [FPDProgressiveRenderNew](#) , [FPDProgressiveRenderDestroy](#) .

### Returned from

[FPDProgressiveRenderNew](#)

### Used by

[FPDProgressiveRenderClear](#)  
[FPDProgressiveRenderContinue](#)  
[FPDProgressiveRenderDestroy](#)  
[FPDProgressiveRenderEstimateProgress](#)  
[FPDProgressiveRenderStart](#)

## Functions

### Functions summary

[FPDProgressiveRenderClear](#)

Gets ready for next rendering.

[FPDProgressiveRenderContinue](#)

Continue rendering.

[FPDProgressiveRenderDestroy](#)

Destroys the PDF progressive renderer object.

[FPDProgressiveRenderEstimateProgress](#)

Estimates percentage of progress.

[FPDProgressiveRenderNew](#)

Creates a new empty PDF progressive renderer object.

**[FPDProgressiveRenderStart](#)**

Starts rendering.

**Functions detail****[FPDProgressiveRenderClear](#)****Syntax**

```
void FPDProgressiveRenderClear (
    FPD\_ProgressiveRender render
);
```

**Description**

Gets ready for next rendering.

**Parameter**

---

render	[In] The input PDF progressive renderer object.
--------	---

---

**Return**

void

**Head file reference**

[fpd\\_renderTempl.h](#): 402

**[FPDProgressiveRenderContinue](#)****Syntax**

```
void FPDProgressiveRenderContinue (
    FPD\_ProgressiveRender render,
    FS\_PauseHandler pauseHandler
);
```

**Description**

Continue rendering.

**Parameter**

---

render	[In] The input PDF progressive renderer object.
--------	---

---

pauseHandler	[In] The pause handler.
--------------	-------------------------

---

**Return**

void

**Head file reference**

[fpd\\_renderTempl.h](#): 383

## FPDProgressiveRenderDestroy

### Syntax

```
void FPDProgressiveRenderDestroy (
    FPD\_ProgressiveRender render
);
```

### Description

Destroys the PDF progressive renderer object.

### Parameter

---

render	[In] The input PDF progressive renderer object.
--------	---

---

### Return

void

### Head file reference

fpd\_renderTempl.h: 361

## FPDProgressiveRenderEstimateProgress

### Syntax

```
FS_INT32 FPDProgressiveRenderEstimateProgress (
    FPD\_ProgressiveRender render
);
```

### Description

Estimates percentage of progress.

### Parameter

---

render	[In] The input PDF progressive renderer object.
--------	---

---

### Return

An estimated percentage of progress.

### Head file reference

fpd\_renderTempl.h: 393

## FPDProgressiveRenderNew

### Syntax

```
FPD_ProgressiveRender FPDProgressiveRenderNew (void );
```

**Description**

Creates a new empty PDF progressive renderer object.

**Return**

A new empty PDF progressive renderer object.

**Head file reference**

fpd\_renderTempl.h: 352

## FPDProgressiveRenderStart

**Syntax**

```
void FPDProgressiveRenderStart (
    FPD\_ProgressiveRender render,
    FPD\_RenderContext context,
    FPD\_RenderDevice* device,
    const FPD\_RenderOptions options,
    FS\_PauseHandler pauseHandler
);
```

**Description**

Starts rendering.

**Parameter**

---

render	[In] The input PDF progressive renderer object.
--------	---

---

context	[In] The render context.
---------	--------------------------

---

device	[In] The output device.
--------	-------------------------

---

options	[In] The render options.
---------	--------------------------

---

pauseHandler	[In] The pause handler.
--------------	-------------------------

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 370

## FPD\_ProgressiveSearch

[Return from](#) [Used by](#)

## Description

A progressive search facility for stream-based text searching inside a single page. See [FPDProgressiveSearchNew](#) , [FPDProgressiveSearchDestroy](#) .

### Returned from

[FPDProgressiveSearchNew](#)

### Used by

[FPDProgressiveSearchContinue](#)  
[FPDProgressiveSearchCountRects](#)  
[FPDProgressiveSearchDestroy](#)  
[FPDProgressiveSearchFindFrom](#)  
[FPDProgressiveSearchFindNext](#)  
[FPDProgressiveSearchFindPrev](#)  
[FPDProgressiveSearchGetPos](#)  
[FPDProgressiveSearchGetRect](#)  
[FPDProgressiveSearchGetStatus](#)

## Definitions

### Definitions summary

#### [FPD\\_TEXT\\_MATCHCASE](#)

Whether to match the character case(upper/lower).

#### [FPDTEXT\\_CONSECUTIVE](#)

Whether to do consecutive searching.

#### [FPDTEXT\\_MATCHWHOLEWORD](#)

Whether to match the whole word for the search pattern.

#### [FPD\\_SCH\\_STATUS\\_FAILED](#)

Failed.

#### [FPD\\_SCH\\_STATUS\\_FOUND](#)

Found.

#### [FPD\\_SCH\\_STATUS\\_NOTFOUND](#)

Not found.

#### [FPD\\_SCH\\_STATUS\\_READY](#)

Ready.

#### [FPD\\_SCH\\_STATUS\\_TOBECONTINUED](#)

To be continued.

### Definitions detail

#### FPD\_TEXT\_MATCHCASE

##### Syntax

```
#define FPD_TEXT_MATCHCASE 0x00000001
```

##### Description

Whether to match the character case(upper/lower).

**Group**[FPDSearchFlags](#)**Head file reference**

fpd\_textExpT.h: 84

**FPDTEXT\_CONSECUTIVE****Syntax**

#define FPDTEXT\_CONSECUTIVE 0x00000004

**Description**

Whether to do consecutive searching.

**Group**[FPDSearchFlags](#)**Head file reference**

fpd\_textExpT.h: 90

**FPDTEXT\_MATCHWHOLEWORD****Syntax**

#define FPDTEXT\_MATCHWHOLEWORD 0x00000002

**Description**

Whether to match the whole word for the search pattern.

**Group**[FPDSearchFlags](#)**Head file reference**

fpd\_textExpT.h: 87

**FPD\_SCH\_STATUS\_FAILED****Syntax**

#define FPD\_SCH\_STATUS\_FAILED 4

**Description**

Failed.

**Group**[FPDSearchingStatus](#)**Head file reference**

fpd\_textExpT.h: 72

## FPD\_SCH\_STATUS\_FOUND

### Syntax

#define FPD\_SCH\_STATUS\_FOUND 2

### Description

Found.

### Group

[FPDSearchingStatus](#)

### Head file reference

fpd\_textExpT.h: 68

## FPD\_SCH\_STATUS\_NOTFOUND

### Syntax

#define FPD\_SCH\_STATUS\_NOTFOUND 3

### Description

Not found.

### Group

[FPDSearchingStatus](#)

### Head file reference

fpd\_textExpT.h: 70

## FPD\_SCH\_STATUS\_READY

### Syntax

#define FPD\_SCH\_STATUS\_READY 0

### Description

Ready.

### Group

[FPDSearchingStatus](#)

### Head file reference

fpd\_textExpT.h: 64

## FPD\_SCH\_STATUS\_TOBECONTINUED

**Syntax**

```
#define FPD_SCH_STATUS_TOBECONTINUED 1
```

**Description**

To be continued.

**Group**

[FPDSearchingStatus](#)

**Head file reference**

fpd\_textExpT.h: 66

## Functions

### Functions summary

**[FPDProgressiveSearchContinue](#)**

Continues the searching.

**[FPDProgressiveSearchCountRects](#)**

Gets the count of rectangles for current found matches.

**[FPDProgressiveSearchDestroy](#)**

Destroys the progressive searching object.

**[FPDProgressiveSearchFindFrom](#)**

Find the first match with specified search flags, optionally with a start position. The page object may have parsed content objects, or may not have those objects parsed. In later case, the search engine will do a text-only parsing in order to search for the pattern. This function can be paused by a pause object.

**[FPDProgressiveSearchFindNext](#)**

Searchs for next match within the same page, from last match position.

**[FPDProgressiveSearchFindPrev](#)**

Searchs for previous match within the same page, from last match position.

**[FPDProgressiveSearchGetPos](#)**

Gets the current position.

**[FPDProgressiveSearchGetRect](#)**

Gets the specified rectangle for current found matches.

**[FPDProgressiveSearchGetStatus](#)**

Gets the current status.

**[FPDProgressiveSearchNew](#)**

Creates a progressive searching object.

### Functions detail

**FPDProgressiveSearchContinue****Syntax**

```
void FPDProgressiveSearchContinue (
    FPD\_ProgressiveSearch sch
);
```

**Description**

Continues the searching.

#### Parameter

---

sch	[In] The input progressive searching object.
-----	--

---

#### Return

void

#### Head file reference

fpd\_textTempl.h: 65

### FPDProgressiveSearchCountRects

#### Syntax

```
FS_DWORD FPDProgressiveSearchCountRects (
    FPD\_ProgressiveSearch sch
);
```

#### Description

Gets the count of rectangles for current found matches.

#### Parameter

---

sch	[In] The input progressive searching object.
-----	--

---

#### Return

The count of rectangles for current found matches.

#### Head file reference

fpd\_textTempl.h: 92

**Note:** Before called, the page must be parsed.

### FPDProgressiveSearchDestroy

#### Syntax

```
void FPDProgressiveSearchDestroy (
    FPD\_ProgressiveSearch sch
);
```

#### Description

Destroys the progressive searching object.

#### Parameter

---

sch	[In] The input progressive searching object.
-----	--

---

**Return**

void

**Head file reference**

fpd\_textTempl.h: 29

**FPDProgressiveSearchFindFrom****Syntax**

```
void FPDProgressiveSearchFindFrom (
    FPD\_ProgressiveSearch sch,
    FPD\_Page page,
    FS\_LPCWSTR pattern,
    FS\_INT32 pos,
    FS\_DWORD flags,
    FS\_PauseHandler pauseHandler
);
```

**Description**

Find the first match with specified search flags, optionally with a start position. The page object may have parsed content objects, or may not have those objects parsed. In later case, the search engine will do a text-only parsing in order to search for the pattern. This function can be paused by a pause object.

**Parameter**


---

sch	[In] The input progressive searching object.
-----	--

---

page	[In] A PDF page handle.
------	-------------------------

---

pattern	[In] What do you want to find?
---------	--------------------------------

---

pos	[In] The starting position.
-----	-----------------------------

---

flags	[In] See <a href="#">FPDSearchFlags</a> .
-------	---

---

pauseHandler	[In] The user-supplied pause handler that can pause the finding process. Can be NULL if no pausing is needed.
--------------	---

---

**Return**

void

**Head file reference**

fpd\_textTempl.h: 47

## FPDProgressiveSearchFindNext

### Syntax

```
void FPDProgressiveSearchFindNext (
    FPD\_ProgressiveSearch sch
);
```

### Description

Searchs for next match within the same page, from last match position.

### Parameter

---

sch	[In] The input progressive searching object.
-----	--

---

### Return

void

### Head file reference

fpd\_textTempl.h: 74

## FPDProgressiveSearchFindPrev

### Syntax

```
void FPDProgressiveSearchFindPrev (
    FPD\_ProgressiveSearch sch
);
```

### Description

Searchs for previous match within the same page, from last match position.

### Parameter

---

sch	[In] The input progressive searching object.
-----	--

---

### Return

void

### Head file reference

fpd\_textTempl.h: 83

## FPDProgressiveSearchGetPos

### Syntax

```
FS_INT32 FPDProgressiveSearchGetPos (
    FPD\_ProgressiveSearch sch
);
```

**Description**

Gets the current position.

**Parameter**


---

sch	[In] The input progressive searching object.
-----	--

---

**Return**

The current position.

**Head file reference**

fpd\_textTempl.h: 111

**FPDProgressiveSearchGetRect****Syntax**

```
FS_FloatRect FPDProgressiveSearchGetRect (
    FPD\_ProgressiveSearch sch,
    FS\_INT32 index
);
```

**Description**

Gets the specified rectangle for current found matches.

**Parameter**


---

sch	[In] The input progressive searching object.
-----	--

---

index	[In] The index of the current found matches.
-------	--

---

**Return**

The specified rectangle for current found matches.

**Head file reference**

fpd\_textTempl.h: 101

**Note:** Before called, the page must be parsed.

**FPDProgressiveSearchGetStatus****Syntax**

```
FS_INT32 FPDProgressiveSearchGetStatus (
    FPD\_ProgressiveSearch sch
);
```

**Description**

Gets the current status.

**Parameter**

---

sch	[In] The input progressive searching object.
-----	--

---

**Return**

The current status.

**Head file reference**

fpd\_textTempl.h: 38

**FPDProgressiveSearchNew****Syntax**

FPD\_ProgressiveSearch FPDProgressiveSearchNew (void );

**Description**

Creates a progressive searching object.

**Return**

A progressive searching object.

**Head file reference**

fpd\_textTempl.h: 21

# FPD\_Reference

## Description

A FPD\_Reference is a additional type of object:

Objects may be labeled so that they can be referred to by other objects. A labeled object is called an indirect object.

## Functions

### Functions summary

**[FPDReferenceGetRefObjNum](#)**

Gets the referred object number.

**[FPDReferenceIdentical](#)**

Compares with another object.

**[FPDReferenceNew](#)**

Creates a reference object with indirect object collection and referred object number.

**[FPDReferenceNew2](#)**

Creates a reference object with indirect object collection and referred object number.

**FPDReferenceSetRefToDoc**

Changes the reference.

**FPDReferenceSetRefToFDFDoc**

Changes the reference.

## Functions detail

### FPDReferenceGetRefObjNum

**Syntax**

```
FS_DWORD FPDReferenceGetRefObjNum (
    FPD Object objReference
);
```

**Description**

Gets the referred object number.

**Parameter**

---

objReference	[In] The input reference object.
--------------	----------------------------------

---

**Return**

The referred object number.

**Head file reference**

fpd\_objsTempl.h: 1482

### FPDReferenceIdentical

**Syntax**

```
FS_BOOL FPDReferenceIdentical (
    FPD Object objReference,
    FPD Object otherReference
);
```

**Description**

Compares with another object.

**Parameter**

---

objReference	[In] The input reference object.
--------------	----------------------------------

---

otherReference	[In] The another reference object.
----------------	------------------------------------

---

**Return**

Non-zero means identical, otherwise not identical.

**Head file reference**

fpd\_objsTempl.h: 1513

**FPDReferenceNew****Syntax**

```
FPD_Object FPDReferenceNew (
    FPD Document doc,
    FS\_INT32 objNum
);
```

**Description**

Creates a reference object with indirect object collection and referred object number.

**Parameter**

---

doc	[In] The indirect object collection.
-----	--------------------------------------

---

objNum	[In] The referred object number.
--------	----------------------------------

---

**Return**

A reference object.

**Head file reference**

fpd\_objsTempl.h: 1462

**FPDReferenceNew2****Syntax**

```
FPD_Object FPDReferenceNew2 (
    FDF Document doc,
    FS\_INT32 objNum
);
```

**Description**

Creates a reference object with indirect object collection and referred object number.

**Parameter**

---

doc	[In] The indirect object collection.
-----	--------------------------------------

---

objNum	[In] The referred object number.
--------	----------------------------------

---

**Return**

A reference object.

**Head file reference**

fpd\_objsTempl.h: 1472

**FPDReferenceSetRefToDoc****Syntax**

```
void FPDReferenceSetRefToDoc (
    FPD Object objReference,
    FPD Document doc,
    FS DWORD objNum
);
```

**Description**

Changes the reference.

**Parameter**

---

objReference	[In] The input reference object.
--------------	----------------------------------

---

doc	[In] The new indirect object collection.
-----	--

---

objNum	[In] The new referred indirect object number.
--------	---

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1491

**FPDReferenceSetRefToFDFDoc****Syntax**

```
void FPDReferenceSetRefToFDFDoc (
    FPD Object objReference,
    FDF Document doc,
    FS DWORD objNum
);
```

**Description**

Changes the reference.

**Parameter**

---

objReference	[In] The input reference object.
--------------	----------------------------------

---

---

doc	[In] The new indirect object collection.
-----	--

---

objNum	[In] The new referred indirect object number.
--------	---

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1502

## FPD\_RenderContext

**Return from Used by**

### Description

Context for rendering a PDF page or a list of page objects. A PDF page can be divided into different layers, including the page content, annotations, and interactive form. It's important to keep information about those layers if backdrop generation is required during transparency rendering for devices not capable of fetching background. A PDF rendering context also makes use of page caches, additional page resources, and rendering options. See [FPDRenderContextNew](#) , [FPDRenderContextNew2](#) , [FPDRenderContextDestroy](#) .

### Returned from

[FPDRenderContextNew](#)  
[FPDRenderContextNew2](#)

### Used by

[FPDAnotListDisplayAnnotsEx](#)  
[FPDProgressiveRenderStart](#)  
[FPDRenderDeviceDrawType3Text](#)  
[FPDRenderContextAppendForm](#)  
[FPDRenderContextAppendPage](#)  
[FPDRenderContextDestroy](#)  
[FPDRenderContextDrawForm](#)  
[FPDRenderContextDrawPage](#)  
[FPDRenderContextDrawStream](#)  
[FPDRenderContextGetBackground](#)  
[FPDRenderContextGetPageCache](#)  
[FPDRenderContextQuickDraw](#)  
[FPDRenderContextRender](#)  
[FPDRenderContextSetBackground](#)

## Functions

### Functions summary

[FPDRenderContextAppendForm](#)

Append Form to the current layer.

**[FPDRenderContextAppendPage](#)**

Append page to the current layer.

**[FPDRenderContextCreateBackgroundDrawHandler](#)**

Creates the background drawing handler.

**[FPDRenderContextDeleteBackgroundDrawHandler](#)**

Deletes the background drawing handler.

**[FPDRenderContextDestroy](#)**

Destroys the PDF rendering context object.

**[FPDRenderContextDrawForm](#)**

Draws a list of page objects.

**[FPDRenderContextDrawPage](#)**

Draws a list of page objects.

**[FPDRenderContextDrawStream](#)**

Draws a page description stream.

**[FPDRenderContextGetBackground](#)**

Gets background of a page object within the rendering context.

**[FPDRenderContextGetPageCache](#)**

Gets the page render cache.

**[FPDRenderContextNew](#)**

Creates a new PDF rendering context object.

**[FPDRenderContextNew2](#)**

Creates a new PDF rendering context object.

**[FPDRenderContextQuickDraw](#)**

Draws a rough preview (quick draw).

**[FPDRenderContextRender](#)**

Do the real rendering. Optionally, rendering matrix can be modified by the last matrix.

**[FPDRenderContextSetBackground](#)**

Sets custom background drawing.

## Functions detail

### **FPDRenderContextAppendForm**

#### **Description**

Append Form to the current layer.

#### **Parameter**

---

context	[In] The input PDF rendering context object.
---------	--

---

objs	[In] The input Form
------	---------------------

---

---

object2Device	[In] The matrix from object coords to device coords
---------------	---

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 235

**FPDRenderContextAppendPage****Syntax**

```
void FPDRenderContextAppendPage (
    FPD\_RenderContext context,
    FPD\_Page objs,
    FS\_AffineMatrix object2Device
);
```

**Description**

Append page to the current layer.

**Parameter**


---

context	[In] The input PDF rendering context object.
---------	--

---

objs	[In] The input page.
------	----------------------

---

object2Device	[In] The matrix from object coords to device coords.
---------------	--

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 224

**FPDRenderContextCreateBackgroundDrawHandler****Syntax**

```
FPD_BackgroundDrawHandler FPDRenderContextCreateBackgroundDrawHandler (
    FPD\_BackgroundDraw backgroundDraw
);
```

**Description**

Creates the background drawing handler.

**Parameter**

---

backgroundDraw	[In] The input background drawing handler structure.
----------------	--

---

**Return**

The background drawing handler.

**Head file reference**

fpd\_renderTempl.h: 256

**FPDRenderContextDeleteBackgroundDrawHandler****Syntax**

```
void FPDRenderContextDeleteBackgroundDrawHandler (
    FPD\_BackgroundDrawHandler backgroundDrawHandler
);
```

**Description**

Deletes the background drawing handler.

**Parameter**

---

backgroundDrawHandler	[In] The input background drawing handler structure.
-----------------------	--

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 265

**FPDRenderContextDestroy****Syntax**

```
void FPDRenderContextDestroy (
    FPD\_RenderContext context
);
```

**Description**

Destroys the PDF rendering context object.

**Parameter**

---

context	[In] The input PDF rendering context object.
---------	--

---

**Return**

void

**Head file reference**

---

fpd\_renderTempl.h: 202

## FPDRenderContextDrawForm

### Syntax

```
void FPDRenderContextDrawForm (
    FPD\_RenderContext context,
    FPD\_RenderDevice* outDevice,
    FPD\_Form objs,
    FS\_AffineMatrix* pObject2Device,
    const FPD\_RenderOptions options
);
```

### Description

Draws a list of page objects.

### Parameter

context	[In] The input PDF rendering context object.
outDevice	[Out] The output device.
objs	[In] The input Form.
pObject2Device	[In] The matrix from object coords to device coords, can be NULL.
options	[In] The render options.

### Return

void

### Head file reference

fpd\_renderTempl.h: 299

## FPDRenderContextDrawPage

### Syntax

```
void FPDRenderContextDrawPage (
    FPD\_RenderContext context,
    FPD\_RenderDevice* outDevice,
    FPD\_Page objs,
    FS\_AffineMatrix* pObject2Device,
    const FPD\_RenderOptions options
);
```

### Description

Draws a list of page objects.

**Parameter**

context	[In] The input PDF rendering context object.
outDevice	[Out] The output device.
objs	[In] The input page.
pObject2Device	[In] The matrix from object coords to device coords, can be NULL.
options	[In] The render options.

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 286

**FPDRenderContextDrawStream****Syntax**

```
void FPDRenderContextDrawStream (
    FPD\_RenderContext context,
    FPD\_RenderDevice* outDevice,
    const void* pStreamDataBuf,
    FS\_DWORD stream_size,
    FS\_AffineMatrix displayMatrix
);
```

**Description**

Draws a page description stream.

**Parameter**

context	[In] The input PDF rendering context object.
outDevice	[Out] The output device.
pStreamDataBuf	[In] The page description stream buffer.

---

stream_size	[In] The size in bytes of the page description stream. -1 for <a href="#">NULL</a> terminated byte string.
-------------	--

---

displayMatrix	[In] The matrix from stream coords to device coords. Can be <a href="#">NULL</a> if no transformation needed.
---------------	---

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 211

**FPDRenderContextGetBackground****Syntax**

```
void FPDRenderContextGetBackground (
    FPD\_RenderContext context,
    FS\_DIBitmap* outBitmapBuffer,
    const FPD\_PageObject obj,
    const FPD\_RenderOptions options,
    FS\_AffineMatrix* pFinalMatrix
);
```

**Description**

Gets background of a page object within the rendering context.

**Parameter**


---

context	[In] The input PDF rendering context object.
---------	--

---

outBitmapBuffer	[Out] It receives the background bitmap.
-----------------	--

---

obj	[In] The input page object.
-----	-----------------------------

---

options	[In] The render options.
---------	--------------------------

---

pFinalMatrix	[In] The matrix from object coords to device coords.
--------------	--

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 322

## FPDRenderContextGetPageCache

### Syntax

```
FPD_PageRenderCache FPDRenderContextGetPageCache (
    FPD\_RenderContext context
);
```

### Description

Gets the page render cache.

### Parameter

---

context	[In] The input PDF rendering context object.
---------	--

---

### Return

The page render cache.

### Head file reference

[fpd\\_renderTempl.h](#): 335

## FPDRenderContextNew

### Syntax

```
FPD_RenderContext FPDRenderContextNew (
    FPD\_Page page,
    FS\_BOOL bFirstLayer
);
```

### Description

Creates a new PDF rendering context object.

### Parameter

---

page	[In] The input page.
------	----------------------

---

---

bFirstLayer	[In] Whether it's the first layer.
-------------	------------------------------------

---

### Return

A empty PDF rendering context object.

### Head file reference

[fpd\\_renderTempl.h](#): 180

## FPDRenderContextNew2

### Syntax



```
FPD_RenderContext FPDRenderContextNew2 (
    FPD Document doc,
    FPD PageRenderCache pageCache,
    FPD Object pageResources,
    FS\_BOOL bFirstLayer
);
```

**Description**

Creates a new PDF rendering context object.

**Parameter**

---

doc	[In] The PDF document.
-----	------------------------

---

pageCache	[In] The page render cache.
-----------	-----------------------------

---

pageResources	[In] The Resources dictionary.
---------------	--------------------------------

---

bFirstLayer	[In] Whether it's the first layer.
-------------	------------------------------------

---

**Return**

A new PDF rendering context object.

**Head file reference**

fpd\_renderTempl.h: 190

**FPDRenderContextQuickDraw****Syntax**

```
void FPDRenderContextQuickDraw (
    FPD RenderContext context,
    FPD RenderDevice* outDevice
);
```

**Description**

Draws a rough preview (quick draw).

**Parameter**

---

context	[In] The input PDF rendering context object.
---------	--

---

outDevice	[Out] The output device.
-----------	--------------------------

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 312

**FPDRenderContextRender****Syntax**

```
void FPDRenderContextRender (
    FPD\_RenderContext context,
    FPD\_RenderDevice device,
    const FPD\_RenderOptions options,
    FS\_AffineMatrix* pFinalMatrix
);
```

**Description**

Do the real rendering. Optionally, rendering matrix can be modified by the last matrix.

**Parameter**

context	[In] The input PDF rendering context object.
device	[In] The output device.
options	[In] The render options.
pFinalMatrix	[In] The final matrix to transform the result, Default value is <a href="#">NULL</a> .

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 274

**FPDRenderContextSetBackground****Syntax**

```
void FPDRenderContextSetBackground (
    FPD\_RenderContext context,
    FPD\_BackgroundDrawHandler BackgroundDrawHandler
);
```

**Description**

Sets custom background drawing.

**Parameter**

---

context	[In] The input PDF rendering context object.
---------	--

---

BackgroundDrawHandler	[In] The background drawing handler.
-----------------------	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 246

## FPD\_RenderDevice

**[Return from Used by](#)**

### Description

The base function set for render devices. This set provide all rendering features common to all output devices (including display, printer, and special devices like converters). The render device makes use of Foxit Rendering Device Driver to do the actual output. See [FPDRenderDeviceNew](#) , [FPDRenderDeviceDestroy](#) .

### Returned from

[FPDFxgeDeviceNew](#)  
[FPDWindowsDeviceNew](#)  
[FPDRenderDeviceNew](#)

### Used by

[FPDAnotDrawAppearance](#)  
[FPDAnotDrawBorder](#)  
[FPDAnotListDisplayAnnots](#)  
[FPDFFormControlDrawControl](#)  
[FPDFxgeDeviceAttach](#)  
[FPDFxgeDeviceCreate](#)  
[FPDFxgeDeviceDestroy](#)  
[FPDProgressiveRenderStart](#)  
[FPDRenderContextDrawForm](#)  
[FPDRenderContextDrawPage](#)  
[FPDRenderContextDrawStream](#)  
[FPDRenderContextQuickDraw](#)  
[FPDRenderContextRender](#)  
[FPDWindowsDeviceDestroy](#)  
[FPDRenderDeviceCreateCompatibleBitmap](#)  
[FPDRenderDeviceDestroy](#)  
[FPDRenderDeviceDrawCosmeticLine](#)  
[FPDRenderDeviceDrawNormalText](#)  
[FPDRenderDeviceDrawPath](#)  
[FPDRenderDeviceDrawTextPath](#)  
[FPDRenderDeviceDrawTextString](#)  
[FPDRenderDeviceDrawTextString2](#)

[\*\*FPDRenderDeviceDrawType3Text\*\*](#)  
[\*\*FPDRenderDeviceEndRendering\*\*](#)  
[\*\*FPDRenderDeviceFillRect\*\*](#)  
[\*\*FPDRenderDeviceGetBitmap\*\*](#)  
[\*\*FPDRenderDeviceGetBPP\*\*](#)  
[\*\*FPDRenderDeviceGetDeviceCapsXY\*\*](#)  
[\*\*FPDRenderDeviceGetDIBits\*\*](#)  
[\*\*FPDRenderDeviceGetDitherBits\*\*](#)  
[\*\*FPDRenderDeviceGetHeight\*\*](#)  
[\*\*FPDRenderDeviceGetRenderCaps\*\*](#)  
[\*\*FPDRenderDeviceGetWidth\*\*](#)  
[\*\*FPDRenderDeviceRestoreState\*\*](#)  
[\*\*FPDRenderDeviceSaveState\*\*](#)  
[\*\*FPDRenderDeviceSetBitmap\*\*](#)  
[\*\*FPDRenderDeviceSetBitMask\*\*](#)  
[\*\*FPDRenderDeviceSetClip\\_Rect\*\*](#)  
[\*\*FPDRenderDeviceSetDIBits\*\*](#)  
[\*\*FPDRenderDeviceSetPixel\*\*](#)  
[\*\*FPDRenderDeviceStartRendering\*\*](#)

## Functions

### Functions summary

[\*\*FPDRenderDeviceCreateCompatibleBitmap\*\*](#)

Creates a compatible bitmap.

[\*\*FPDRenderDeviceDestroy\*\*](#)

Destroys the render device object.

[\*\*FPDRenderDeviceDrawCosmeticLine\*\*](#)

Draws a single pixel (device dependent) line.

[\*\*FPDRenderDeviceDrawNormalText\*\*](#)

Draws normal text.

[\*\*FPDRenderDeviceDrawPath\*\*](#)

Draw a path. If either *fill\_argb* or *stroke\_argb* is used and with alpha value between 0 and 255, then device capability FSRC\_ALPHA\_PATH or FSRC\_GET\_BITS is required.

[\*\*FPDRenderDeviceDrawTextPath\*\*](#)

Draws text path.

[\*\*FPDRenderDeviceDrawTextString\*\*](#)

Draws a text string, using Windows style parameters.

[\*\*FPDRenderDeviceDrawTextString2\*\*](#)

Draws a text string using PDF style parameters.

[\*\*FPDRenderDeviceDrawType3Text\*\*](#)

Draws type3 text.

[\*\*FPDRenderDeviceEndRendering\*\*](#)

End rendering.

[\*\*FPDRenderDeviceFillRect\*\*](#)

Fills a rectangle.

[\*\*FPDRenderDeviceGetBitmap\*\*](#)

Gets the bitmap of the device.

[\*\*FPDRenderDeviceGetBPP\*\*](#)

Gets the bits per pixel.

[\*\*FPDRenderDeviceGetDeviceCapsXY\*\*](#)

Gets the render capabilities.

**[FPDRenderDeviceGetDIBits](#)**

Loads device buffer into a DIB.

**[FPDRenderDeviceGetDitherBits](#)**

Gets the dither bits.

**[FPDRenderDeviceGetHeight](#)**

Gets the device height.

**[FPDRenderDeviceGetRenderCaps](#)**

Gets the render capabilities.

**[FPDRenderDeviceGetWidth](#)**

Gets the device width.

**[FPDRenderDeviceNew](#)**

Creates a new empty render device object.

**[FPDRenderDeviceRestoreState](#)**

Restores all graphic states.

**[FPDRenderDeviceSaveState](#)**

Saves all graphic states.

**[FPDRenderDeviceSetBitmap](#)**

Sets the bitmap to the device.

**[FPDRenderDeviceSetBitMask](#)**

Outputs masked bitmap. The bitmap can be a monochrome bitmask, or a 8-bit alpha mask.

**[FPDRenderDeviceSetClip\\_PathFill](#)**

Set clipping path using filled region.

**[FPDRenderDeviceSetClip\\_Rect](#)**

Sets a clipping rectangle.

**[FPDRenderDeviceSetDIBits](#)**

Outputs a colored DIB, pixel-to-pixel.

**[FPDRenderDeviceSetPixel](#)**

Sets or composite a pixel.

**[FPDRenderDeviceStartRendering](#)**

Starts rendering.

**[FPDRenderDeviceStretchBitMask](#)**

Outputs masked bitmap. The bitmap can be a monochrome bitmask, or a 8-bit alpha mask.

**[FPDRenderDeviceStretchDIBits](#)**

Stretches a colored DIB onto the device.

## Functions detail

### **[FPDRenderDeviceCreateCompatibleBitmap](#)**

#### **Syntax**

```
void FPDRenderDeviceCreateCompatibleBitmap (
    FPD\_RenderDevice dc,
    FS\_DIBitmap* inoutDIB,
    FS\_INT32 width,
    FS\_INT32 height
);
```

#### **Description**

Creates a compatible bitmap.

#### **Parameter**

---

dc	[In] The input render device object.
inoutDIB	[In/Out] It receives the created bitmap data.
width	[In/Out] The bitmap width.
height	[In/Out] The bitmap height.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 561

**FPDRenderDeviceDestroy****Syntax**

```
void FPDRenderDeviceDestroy (
    FPD\_RenderDevice dc
);
```

**Description**

Destroys the render device object.

**Parameter**


---

dc	[In] The input render device object.
----	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 428

**FPDRenderDeviceDrawCosmeticLine****Syntax**

```
FS_BOOL FPDRenderDeviceDrawCosmeticLine (
    FPD\_RenderDevice dc,
    FS\_FLOAT x1,
    FS\_FLOAT y1,
    FS\_FLOAT x2,
    FS\_FLOAT y2,
    FS\_ARGB argb
);
```

**Description**

Draws a single pixel (device dependent) line.

**Parameter**

dc	[In] The input render device object.
x1	[In] The x-coordinate of the start point.
y1	[In] The y-coordinate of the start point.
x2	[In] The x-coordinate of the end point.
y2	[In] The y-coordinate of the end point.
argb	[In] The line color.

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 584

**FPDRenderDeviceDrawNormalText****Syntax**

```
FS_BOOL FPDRenderDeviceDrawNormalText (
    FPD_RenderDevice dc,
    FS_INT32 nChars,
    FS_DWORD* pCharCodesBuf,
    FS_FLOAT* pCharPosBuf,
    FPD_Font font,
    FS_FLOAT fontSize,
    FS_AffineMatrix text2Device,
    FS_ARGB fillArgb,
    const FPD_RenderOptions opts
);
```

**Description**

Draws normal text.

**Parameter**

dc	[In] The input render device object.
----	--------------------------------------



---

nChars	[In] The number of characters in the text.
pCharCodesBuf	[In] The character codes.
pCharPosBuf	[In] The character positions.
font	[In] The font will be used to draw text.
fontSize	[In] The font size.
text2Device	[In] The matrix from text coordinate to device coordinate.
fillArgb	[In] The fill color used to fill the text.
opts	[In] The render options.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 736

**FPDRenderDeviceDrawPath****Syntax**

```
FS_BOOL FPDRenderDeviceDrawPath (
    FPD\_RenderDevice dc,
    FPD\_Path path,
    FS\_AffineMatrix pObject2Device,
    FPD\_GraphState graphState,
    FS\_DWORD fill_color,
    FS\_DWORD stroke_color,
    FS\_INT32 fill_mode,
    FS\_BOOL bAntiAlias,
    FS\_INT32 alpha_flag,
    void* pIccTransform
);
```

**Description**

Draw a path. If either *fill\_argb* or *stroke\_argb* is used and with alpha value between 0 and 255, then device capability FSRC\_ALPHA\_PATH or FSRC\_GET\_BITS is required.

**Parameter**

---

dc	[In] The input render device object.
path	[In] Path info.
pObject2Device	[In] Optional transformation.
graphState	[In] Graphic state, for pen attributes.
fill_color	[In] Fill color.
stroke_color	[In] Stroke color.
fill_mode	[In] Fill mode, FSFILL_WINDING or FSFILL_ALTERNATE. 0 for not filled. Also FSFILL_FULLSCREEN or FSFILL_RECT_AA bit can be used with fill mode.
bAntiAlias	[In] Uses anti-alias if the driver supports.
alpha_flag	[In] The flag indicates color type and alpha value, each components 8 bits. alpha_flag == (stroke_alpha<<16) (color_type<
PIccTransform	[In] The color transform handle.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 772

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDRenderDeviceDrawTextPath****Syntax**

```
FS_BOOL FPDRenderDeviceDrawTextPath (
    FPD\_RenderDevice dc,
    FS_INT32 nChars,
    FS_DWORD* pCharCodesBuf,
    FS_FLOAT* pCharPosBuf,
    FPD\_Font font,
    FS_FLOAT fontSize,
    FS\_AffineMatrix text2User,
```

---

```
FS_AffineMatrix user2Device,
const FPD_GraphState graphState,
FS_ARGB fillArgb,
FS_ARGB strokeArgb,
FPD_Path clippingPath
);
```

**Description**

Draws text path.

**Parameter**

dc	[In] The input render device object.
nChars	[In] The number of characters in the text.
pCharCodesBuf	[In] The character codes.
pCharPosBuf	[In] The character positions.
font	[In] The font will be used to draw text.
fontSize	[In] The font size.
text2User	[In] The matrix from text coordinate to user coordinate.
user2Device	[In] The matrix from user coordinate to user coordinate.
graphState	[In] Graphic state, for pen attributes.
fillArgb	[In] Fill color.
strokeArgb	[In] Stroke color
clippingPath	[In] The clipping path to add to.

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 716

**FPDRenderDeviceDrawTextString****Syntax**

```
void FPDRenderDeviceDrawTextString (
    FPD\_RenderDevice dc,
    FS\_INT32 left,
    FS\_INT32 top,
    FPD\_Font font,
    FS\_INT32 height,
    FS\_LPCSTR str,
    FS\_ARGB argb
);
```

**Description**

Draws a text string, using Windows style parameters.

**Parameter**

dc	[In] The input render device object.
----	--------------------------------------

left	[In] x position, in device coordinate.
------	--

top	[In] y position, in device coordinate.
-----	--

font	[In] The input font.
------	----------------------

height	[In] height of the character cell, in pixels.
--------	---

str	[In] a string using font encoding.
-----	------------------------------------

argb	[In] color and alpha value, in Oxaarrggb format.
------	--

**Return**

void

**Head file reference**

[fpd\\_renderTempl.h](#): 679

**FPDRenderDeviceDrawTextString2****Syntax**

```
void FPDRenderDeviceDrawTextString2 (
    FPD\_RenderDevice dc,
    FS\_FLOAT originX,
    FS\_FLOAT originY,
```



---

```
FPD_Font font,
FS_FLOAT fontSize,
FS_AffineMatrix matrix,
FS_ByteString str,
FS_ARGB fillArgb,
FS_ARGB strokeArgb,
const FPD_GraphState graphState,
const FPD_RenderOptions opts
);
```

**Description**

Draws a text string using PDF style parameters.

**Parameter**


---

dc	[In] The input render device object.
originX	[In] x position of the origin (for first character), in device coord.
originY	[In] y position of the origin (for first character), in device coord.
font	[In] The font will be used to draw text.
fontSize	[In] number of points for the font em square.
matrix	[In] a matrix from text space to device space, used for font rotation, scaling and skewing. Can be NULL for identity matrix. If specified, the "e" and "f" coefficients (for translation) are ignored.
str	[In] a string using font encoding.
fillArgb	[In] color and alpha value, in 0xaarrggb format. 0 for not filling.
strokeArgb	[In] color for stroking text. 0 for not stroking.
graphState	[In] required for stroking.
opts	[In] rendering options, like clear-type flag.

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 694

**Since**[SDK LATEEST VERSION > 1.0](#)**FPDRenderDeviceDrawType3Text****Syntax**

```
FS_BOOL FPDRenderDeviceDrawType3Text (
    FPD\_RenderDevice dc,
    FPD\_RenderContext pContext,
    FPD\_RenderOptions pOptions,
    FPD\_PageObject pStates1,
    FS_INT32 nChars,
    FS_DWORD* pCharCodesBuf,
    FS_FLOAT* pCharPosBuf,
    FPD\_Font font,
    FS_FLOAT fontSize,
    FS\_AffineMatrix* pTextToDevice,
    FS\_ARGB fillArgb
);
```

**Description**

Draws type3 text.

**Parameter**

dc	[In] The input render device object.
----	--------------------------------------

pContext	[In] The input render context.
----------	--------------------------------

pOptions	[In] The input render options.
----------	--------------------------------

pStates1	[In] The input graphic states.
----------	--------------------------------

nChars	[In] The number of characters in the text.
--------	--

pCharCodesBuf	[In] The character codes.
---------------	---------------------------

pCharPosBuf	[In] The character positions.
-------------	-------------------------------

font	[In] The Type3 font will be used to draw text.
------	--

fontSize	[In] The font size.
pTextToDevice	[In] The pointer of matrix from text coordinate to device coordinate.
fillArgb	[In] The fill color use to fill the text.

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 753

**FPDRenderDeviceEndRendering****Syntax**

```
void FPDRenderDeviceEndRendering (
    FPD\_RenderDevice dc
);
```

**Description**

End rendering.

**Parameter**

---

dc	[In] The input render device object.
----	--------------------------------------

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 446

**Note:** Called only once for each rendering job.

**FPDRenderDeviceFillRect****Syntax**

```
FS_BOOL FPDRenderDeviceFillRect (
    FPD\_RenderDevice dc,
    const FS\_Rect* pRect,
    FS\_ARGB fill_argb
);
```

**Description**

Fills a rectangle.



**Parameter**

---

dc	[In] The input render device object.
pRect	[In/Out] The input rectangle
fill_argb	[In/Out] The color to fill.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 573

**FPDRenderDeviceGetBitmap****Syntax**

```
FS_DIBitmap FPDRenderDeviceGetBitmap (
    FPD\_RenderDevice dc
);
```

**Description**

Gets the bitmap of the device.

**Parameter**

---

dc	[In] The input render device object.
----	--------------------------------------

---

**Return**

The bitmap of the device.

**Head file reference**

fpd\_renderTempl.h: 520

**FPDRenderDeviceGetBPP****Syntax**

```
FS_BOOL FPDRenderDeviceGetBPP (
    FPD\_RenderDevice dc
);
```

**Description**

Gets the bits per pixel.

**Parameter**

---

dc	[In] The input render device object.
----	--------------------------------------

---

**Return**

The bits per pixel.

**Head file reference**

fpd\_renderTempl.h: 492

**FPDRenderDeviceGetDeviceCapsXY****Syntax**

```
FS_INT32 FPDRenderDeviceGetDeviceCapsXY (
    FPD\_RenderDevice dc,
    FS\_INT32 id
);
```

**Description**

Gets the render capabilities.

**Parameter**

---

dc	[In] The input render device object.
----	--------------------------------------

---

---

id	[In] The input device capability ID
----	-------------------------------------

---

**Return**

The render capabilities.

**Head file reference**

fpd\_renderTempl.h: 510

**FPDRenderDeviceGetDIBits****Syntax**

```
FS_BOOL FPDRenderDeviceGetDIBits (
    FPD\_RenderDevice dc,
    FS\_DIBitmap bitmap,
    FS\_INT32 left,
    FS\_INT32 top
);
```

**Description**

Loads device buffer into a DIB.

**Parameter**

dc	[In] The input render device object.
bitmap	[Out] It receives the loaded device buffer.
left	[In] The x-coordinate in the device.
top	[In] The y-coordinate in the device.

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 598

**FPDRenderDeviceGetDitherBits****Syntax**

```
FS_INT32 FPDRenderDeviceGetDitherBits (
    FPD\_RenderDevice dc
);
```

**Description**

Gets the dither bits.

**Parameter**

dc	[In] The input render device object.
----	--------------------------------------

**Return**

The dither bits.

**Head file reference**

fpd\_renderTempl.h: 670

**FPDRenderDeviceGetHeight****Syntax**

```
FS_INT32 FPDRenderDeviceGetHeight (
    FPD\_RenderDevice dc
);
```

**Description**

Gets the device height.

**Parameter**

---

dc	[In] The input render device object.
----	--------------------------------------

---

**Return**

The device height.

**Head file reference**

fpd\_renderTempl.h: 483

**FPDRenderDeviceGetRenderCaps****Syntax**

```
FS_INT32 FPDRenderDeviceGetRenderCaps (
    FPD\_RenderDevice dc
);
```

**Description**

Gets the render capabilities.

**Parameter**

---

dc	[In] The input render device object.
----	--------------------------------------

---

**Return**

The render capabilities.

**Head file reference**

fpd\_renderTempl.h: 501

**FPDRenderDeviceGetWidth****Syntax**

```
FS_INT32 FPDRenderDeviceGetWidth (
    FPD\_RenderDevice dc
);
```

**Description**

Gets the device width.

**Parameter**

---

dc	[In] The input render device object.
----	--------------------------------------

---

**Return**

The device width.

**Head file reference**

fpd\_renderTempl.h: 474

## FPDRenderDeviceNew

### Syntax

```
FPD_RenderDevice FPDRenderDeviceNew (void );
```

### Description

Creates a new empty render device object.

### Return

A new empty render device object.

### Head file reference

fpd\_renderTempl.h: 419

## FPDRenderDeviceRestoreState

### Syntax

```
void FPDRenderDeviceRestoreState (
    FPD_RenderDevice dc,
    FS_BOOL bKeepSaved
);
```

### Description

Restores all graphic states.

### Parameter

dc	[In] The input render device object.
----	--------------------------------------

bKeepSaved	[In] Whether to keep the restored states saved in buffer.
------------	---

### Return

void

### Head file reference

fpd\_renderTempl.h: 464

## FPDRenderDeviceSaveState

### Syntax

```
void FPDRenderDeviceSaveState (
    FPD_RenderDevice dc
);
```

**Description**

Saves all graphic states.

**Parameter**

---

dc	[In] The input render device object.
----	--------------------------------------

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 455

**FPDRenderDeviceSetBitmap****Syntax**

```
void FPDRenderDeviceSetBitmap (
    FPD\_RenderDevice dc,
    FS\_DIBitmap bitmap
);
```

**Description**

Sets the bitmap to the device.

**Parameter**

---

dc	[In] The input render device object.
----	--------------------------------------

---

---

bitmap	[In] The input bitmap
--------	-----------------------

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 529

**FPDRenderDeviceSetBitMask****Syntax**

```
FS_BOOL FPDRenderDeviceSetBitMask (
    FPD\_RenderDevice dc,
    const FS\_DIBitmap bitmap,
    FS\_INT32 left,
    FS\_INT32 top,
    FS\_ARGB argb
);
```

**Description**

Outputs masked bitmap. The bitmap can be a monochrome bitmask, or a 8-bit alpha mask.

**Parameter**


---

dc	[In] The input render device object.
bitmap	[In] The inpput mask.
left	[In] The x-coordinate in the device.
top	[In] The y-coordinate in the device.
argb	[In] The color to be masked.

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 639

**Note:** If the mask is monochrome, device capability FSRC\_BIT\_MASK or FSRC\_GET\_BITS is required. If the mask is an alpha mask, device capability FSRC\_ALPHA\_MAS>KFSRC\_GET\_BITS is required.

**FPDRenderDeviceSetClip\_PathFill****Syntax**

```
FS_BOOL FPDRenderDeviceSetClip_PathFill (
    dc,
    FPD\_Path path,
    FS\_AffineMatrix pObject2Device,
    FS\_INT32 fill_mode
);
```

**Description**

Set clipping path using filled region.

**Parameter**


---

dc	[In] The input render device object.
path	[In] The input path info.

---

---

pObject2Device	[In] Optional transformation.
----------------	-------------------------------

---

fill_mode	[In] Fill mode, <a href="#">FSFILL_WINDING</a> or <a href="#">FSFILL_ALTERNATE</a> .
-----------	--

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 794

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDRenderDeviceSetClip\_Rect****Syntax**

```
FS_BOOL FPDRenderDeviceSetClip_Rect (
    FPD\_RenderDevice dc,
    const FS\_Rect rect
);
```

**Description**

Sets a clipping rectangle.

**Parameter**


---

dc	[In] The input render device object.
----	--------------------------------------

---

rect	[In] The input clipping rectangle.
------	------------------------------------

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 551

**FPDRenderDeviceSetDIBits****Syntax**

```
FS_BOOL FPDRenderDeviceSetDIBits (
    FPD\_RenderDevice dc,
    const FS\_DIBitmap bitmap,
    FS\_INT32 left,
    FS\_INT32 top,
    FS\_INT32 blend_type
);
```

**Description**

Outputs a colored DIB, pixel-to-pixel.

**Parameter**

dc	[In] The input render device object.
bitmap	[In] The input colored DIB.
left	[In] The x-coordinate in the device.
top	[In] The y-coordinate in the device.
blend_type	[In] Blend mode.

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 610

**Note:** When ARGB bitmap is specified, device capability FSRC\_ALPHA\_IMAGE or FSRC\_GET\_BITS is required. When non-normal blending type is used, device capability FSRC\_BLEND\_MODE or FSRC\_GET\_BITS is required.

**FPDRenderDeviceSetPixel****Syntax**

```
FS_BOOL FPDRenderDeviceSetPixel (
    FPD\_RenderDevice dc,
    FS\_INT32 x,
    FS\_INT32 y,
    FS\_ARGB argb
);
```

**Description**

Sets or composite a pixel.

**Parameter**

dc	[In] The input render device object.
----	--------------------------------------

---

x	[In] The x-coordinate of the pixel.
---	-------------------------------------

---

y	[In] The y-coordinate of the pixel.
---	-------------------------------------

---

argb	[In] The color of the pixel.
------	------------------------------

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 539

**FPDRenderDeviceStartRendering****Syntax**

```
FS_BOOL FPDRenderDeviceStartRendering (
    FPD_RenderDevice dc
);
```

**Description**

Starts rendering.

**Parameter**


---

dc	[In] The input render device object.
----	--------------------------------------

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 437

**Note:** Called only once for each rendering job.

**FPDRenderDeviceStretchBitMask****Syntax**

```
FS_BOOL FPDRenderDeviceStretchBitMask (
    dc,
    bitmap,
    left,
    top,
    dest_width,
    dest_height,
    FS_ARGB argb,
    FS_DWORD flags
);
```

**Description**

Outputs masked bitmap. The bitmap can be a monochrome bitmask, or a 8-bit alpha mask.

**Parameter**


---

dc	[In] The input render device object.
----	--------------------------------------

---

bitmap	[In] The inpput mask.
--------	-----------------------

---

left	[In] The x-coordinate in the device.
------	--------------------------------------

---

top	[In] The y-coordinate in the device.
-----	--------------------------------------

---

dest_width	[In] The destinate width in the device.
------------	---

---

dest_height	[In] The destinate height in the device.
-------------	--

---

argb	[In] The color to be masked.
------	------------------------------

---

flags	[In] The stretching flags.
-------	----------------------------

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 653

**Note:** If the mask is monochrome, device capability FSRC\_BIT\_MASK or FSRC\_GET\_BITS is required. If the mask is an alpha mask, device capability FSRC\_ALPHA\_MASK or FSRC\_GET\_BITS is required.

**FPDRenderDeviceStretchDIBits****Syntax**

```
FS_BOOL FPDRenderDeviceStretchDIBits (
    dc,
    bitmap,
    left,
    top,
    dest_width,
    dest_height,
    FS_DWORD flags
);
```

**Description**

Stretches a colored DIB onto the device.

**Parameter**

dc	[In] The input render device object.
bitmap	[In] The input colored DIB.
left	[In] The x-coordinate in the device.
top	[In] The y-coordinate in the device.
dest_width	[In] The destinate width in the device.
dest_height	[In] The destinate height in the device.
flags	[In] The stretching flags.

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_renderTempl.h: 624

**Note:** When ARGB bitmap is specified, device capability FSRC\_ALPHA\_IMAGE or FSRC\_GET\_BITS is required.

## FPD\_RenderOptions

### [Return from Used by](#)

#### Description

Page rendering options. see CHAPTER 6 in PDF reference. See [FPDRenderOptionsNew](#), [FPDRenderOptionsDestroy](#).

#### Returned from

[FRPageViewGetRenderOptions](#)  
[FPDRenderOptionsNew](#)

#### Used by

[FPDAnotListDisplayAnnots](#)  
[FPDAnotListDisplayAnnotsEx](#)  
[FPDRenderDeviceDrawType3Text](#)  
[FPDRenderOptionsDestroy](#)  
[FPDRenderOptionsGetAddtionalFlag](#)  
[FPDRenderOptionsGetBackColor](#)  
[FPDRenderOptionsGetColorMode](#)  
[FPDRenderOptionsGetForeColor](#)  
[FPDRenderOptionsGetRenderFlag](#)  
[FPDRenderOptionsSetAddtionalFlag](#)  
[FPDRenderOptionsSetBackColor](#)  
[FPDRenderOptionsSetColorMode](#)  
[FPDRenderOptionsSetForeColor](#)  
[FPDRenderOptionsSetOCCHandler](#)  
[FPDRenderOptionsSetRenderFlag](#)  
[FPDRenderOptionsTranslateColor](#)

## Definitions

### Definitions summary

#### [FPD\\_RENDER\\_BGR\\_STRIPE](#)

For clear type: choose BGR stripe device (most device using RGB).

#### [FPD\\_RENDER\\_CLEARTYPE](#)

Use ClearType-like anti-aliasing.

#### [FPD\\_RENDER\\_FORCE\\_DOWNSAMPLE](#)

Always use downsampling for image stretching.

#### [FPD\\_RENDER\\_FORCE\\_HALFTONE](#)

Always use halftone for image stretching.

#### [FPD\\_RENDER\\_LIMITEDIMAGECACHE](#)

Limit image cache size.

#### [FPD\\_RENDER\\_PRINTGRAPHICTEXT](#)

Always output text as graphics (path or bitmap), don't allow device font substitution.

#### [FPD\\_RENDER\\_PRINTPREVIEW](#)

Print preview mode.

#### [FPD\\_RENDER\\_RECT\\_AA](#)

Always use anti-aliasing for rectangle drawing.

#### [FPD\\_RENDER\\_COLOR\\_ALPHA](#)

Output alpha only, ignore color.

#### [FPD\\_RENDER\\_COLOR\\_GRAY](#)

Gray color mode: map colors to background/foreground.

#### [FPD\\_RENDER\\_COLOR\\_NORMAL](#)

Normal color mode.

#### [FPD\\_RENDER\\_COLOR\\_TWOCOLOR](#)

Two color mode: map white/black to background/foreground, other unchanged.

### Definitions detail

#### [FPD\\_RENDER\\_BGR\\_STRIPE](#)

##### Syntax

```
#define FPD_RENDER_BGR_STRIPE 0x10
```

**Description**

For clear type: choose BGR stripe device (most device using RGB).

**Group**

[FPDRenderOptBitmasksFlag](#)

**Head file reference**

fpd\_renderExpT.h: 101

## FPD\_RENDER\_CLEARTYPE

**Syntax**

```
#define FPD_RENDER_CLEARTYPE 0x01
```

**Description**

Use ClearType-like anti-aliasing.

**Group**

[FPDRenderOptBitmasksFlag](#)

**Head file reference**

fpd\_renderExpT.h: 93

## FPD\_RENDER\_FORCE\_DOWNSAMPLE

**Syntax**

```
#define FPD_RENDER_FORCE_DOWNSAMPLE 0x04
```

**Description**

Always use downsampling for image stretching.

**Group**

[FPDRenderOptBitmasksFlag](#)

**Head file reference**

fpd\_renderExpT.h: 97

## FPD\_RENDER\_FORCE\_HALFTONE

**Syntax**

```
#define FPD_RENDER_FORCE_HALFTONE 0x40
```

**Description**

Always use halftone for image stretching.

**Group**

[FPDRenderOptBitmasksFlag](#)**Head file reference**

fpd\_renderExpT.h: 103

**FPD\_RENDER\_LIMITEDIMAGECACHE****Syntax**

#define FPD\_RENDER\_LIMITEDIMAGECACHE 0x80000000

**Description**

Limit image cache size.

**Group**[FPDRenderOptBitmasksFlag](#)**Head file reference**

fpd\_renderExpT.h: 107

**FPD\_RENDER\_PRINTGRAPHICTEXT****Syntax**

#define FPD\_RENDER\_PRINTGRAPHICTEXT 0x02

**Description**

Always output text as graphics (path or bitmap), don't allow device font substitution.

**Group**[FPDRenderOptBitmasksFlag](#)**Head file reference**

fpd\_renderExpT.h: 95

**FPD\_RENDER\_PRINTPREVIEW****Syntax**

#define FPD\_RENDER\_PRINTPREVIEW 0x08

**Description**

Print preview mode.

**Group**[FPDRenderOptBitmasksFlag](#)**Head file reference**

fpd\_renderExpT.h: 99

## FPD\_RENDER\_RECT\_AA

### Syntax

```
#define FPD_RENDER_RECT_AA 0x80
```

### Description

Always use anti-aliasing for rectangle drawing.

### Group

[FPDRenderOptBitmasksFlag](#)

### Head file reference

fpd\_renderExpT.h: 105

## FPD\_RENDER\_COLOR\_ALPHA

### Syntax

```
#define FPD_RENDER_COLOR_ALPHA 3
```

### Description

Output alpha only, ignore color.

### Group

[FPDRenderOptColorModeCodes](#)

### Head file reference

fpd\_renderExpT.h: 80

## FPD\_RENDER\_COLOR\_GRAY

### Syntax

```
#define FPD_RENDER_COLOR_GRAY 1
```

### Description

Gray color mode: map colors to background/foreground.

### Group

[FPDRenderOptColorModeCodes](#)

### Head file reference

fpd\_renderExpT.h: 76

## FPD\_RENDER\_COLOR\_NORMAL

### Syntax

```
#define FPD_RENDER_COLOR_NORMAL 0
```

**Description**

Normal color mode.

**Group**

[FPDRenderOptColorModeCodes](#)

**Head file reference**

fpd\_renderExpT.h: 74

## FPD\_RENDER\_COLOR\_TWOCOLOR

**Syntax**

```
#define FPD_RENDER_COLOR_TWOCOLOR 2
```

**Description**

Two color mode: map white/black to background/foreground, other unchanged.

**Group**

[FPDRenderOptColorModeCodes](#)

**Head file reference**

fpd\_renderExpT.h: 78

## Functions

### Functions summary

[\*\*FPDRenderOptionsCreateOCCContextHandler\*\*](#)

Creates optional content context handler.

[\*\*FPDRenderOptionsDeleteOCCContextHandler\*\*](#)

Deletes optional content context handler.

[\*\*FPDRenderOptionsDestroy\*\*](#)

Destroys the page rendering options object.

[\*\*FPDRenderOptionsGetAdditionalFlag\*\*](#)

Gets additional flags depending on the device.

[\*\*FPDRenderOptionsGetBackColor\*\*](#)

Gets the background color for gray mode (default: white).

[\*\*FPDRenderOptionsGetColorMode\*\*](#)

Gets display mode code.

[\*\*FPDRenderOptionsGetForeColor\*\*](#)

Gets the foreground color for gray mode (default: black).

[\*\*FPDRenderOptionsGetRenderFlag\*\*](#)

Gets render flags.

[\*\*FPDRenderOptionsNew\*\*](#)

Creates a new empty page rendering options object.

[\*\*FPDRenderOptionsSetAdditionalFlag\*\*](#)

Sets additional flags depending on the device.

**[FPDRenderOptionsSetBackColor](#)**

Sets the background color for gray mode (default: white).

**[FPDRenderOptionsSetColorMode](#)**

Sets display mode code.

**[FPDRenderOptionsSetForeColor](#)**

Sets the foreground color for gray mode (default: black).

**[FPDRenderOptionsSetOCCHandler](#)**

Sets optional content context handler.

**[FPDRenderOptionsSetRenderFlag](#)**

Sets render flags.

**[FPDRenderOptionsTranslateColor](#)**

Translates a color.

## Functions detail

**[FPDRenderOptionsCreateOCCHandlerContext](#)****Syntax**

```
FPD_OCCHandlerContext FPDRenderOptionsCreateOCCHandlerContext (
    FPD\_OCCContextCallBack OCContext
);
```

**Description**

Creates optional content context handler.

**Parameter**

---

OCContext	[In] The input page rendering options object.
-----------	---

---

**Return**

FPD\_OCCHandlerContext Input optional content context handler.

**Head file reference**

fpd\_renderTempl.h: 154

**[FPDRenderOptionsDeleteOCCHandlerContext](#)****Syntax**

```
void FPDRenderOptionsDeleteOCCHandlerContext (
    FPD\_OCCHandlerContext OCCHandler
);
```

**Description**

Deletes optional content context handler.

**Parameter**

---

OCCHandler	[In] Input optional content context handler to be deleted.
------------	--

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 163

**FPDRenderOptionsDestroy****Syntax**

```
void FPDRenderOptionsDestroy (
    FPD\_RenderOptions opts
);
```

**Description**

Destroys the page rendering options object.

**Parameter**

---

opts	[In] The input page rendering options object.
------	---

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 30

**FPDRenderOptionsGetAddtionalFlag****Syntax**

```
FS_DWORD FPDRenderOptionsGetAddtionalFlag (
    FPD\_RenderOptions opts
);
```

**Description**

Gets additional flags depending on the device.

**Parameter**

---

opts	[In] The input page rendering options object.
------	---

---

**Return**

The additional flags depending on the device.

**Head file reference**

fpd\_renderTempl.h: 115

## FPDRenderOptionsGetBackColor

### Syntax

```
FS_COLORREF FPDRenderOptionsGetBackColor (
    FPD\_RenderOptions opts
);
```

### Description

Gets the background color for gray mode (default: white).

### Parameter

---

opts	[In] The input page rendering options object.
------	---

---

### Return

The background color for gray mode (default: white).

### Head file reference

fpd\_renderTempl.h: 58

## FPDRenderOptionsGetColorMode

### Syntax

```
FS_INT32 FPDRenderOptionsGetColorMode (
    FPD\_RenderOptions opts
);
```

### Description

Gets display mode code.

### Parameter

---

opts	[In] The input page rendering options object.
------	---

---

### Return

The display mode code.

### Head file reference

fpd\_renderTempl.h: 39

## FPDRenderOptionsGetForeColor

### Syntax

```
FS_COLORREF FPDRenderOptionsGetForeColor (
    FPD\_RenderOptions opts
);
```

**Description**

Gets the foreground color for gray mode (default: black).

**Parameter**

---

opts	[In] The input page rendering options object.
------	---

---

**Return**

The foreground color for gray mode (default: black).

**Head file reference**

fpd\_renderTempl.h: 77

**FPDRenderOptionsGetRenderFlag****Syntax**

```
FS_DWORD FPDRenderOptionsGetRenderFlag (
    FPD\_RenderOptions opts
);
```

**Description**

Gets render flags.

**Parameter**

---

opts	[In] The input page rendering options object.
------	---

---

**Return**

The render flags.

**Head file reference**

fpd\_renderTempl.h: 96

**FPDRenderOptionsNew****Syntax**

```
FPD_RenderOptions FPDRenderOptionsNew (void );
```

**Description**

Creates a new empty page rendering options object.

**Return**

A new empty page rendering options object.

**Head file reference**

fpd\_renderTempl.h: 21

**FPDRenderOptionsSetAdditionalFlag****Syntax**

```
void FPDRenderOptionsSetAdditionalFlag (
    FPD\_RenderOptions opts,
    FS\_DWORD dwFlag
);
```

**Description**

Sets additional flags depending on the device.

**Parameter**

---

opts	[In] The input page rendering options object.
------	---

---

dwFlag	[In] Input additional flags depending on the device.
--------	--

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 124

**FPDRenderOptionsSetBackColor****Syntax**

```
void FPDRenderOptionsSetBackColor (
    FPD\_RenderOptions opts,
    FS\_COLORREF clr
);
```

**Description**

Sets the background color for gray mode (default: white).

**Parameter**

---

opts	[In] The input page rendering options object.
------	---

---

clr	[In] Input the background color for gray mode (default: white).
-----	---

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 67

**FPDRenderOptionsSetColorMode****Syntax**

```
void FPDRenderOptionsSetColorMode (
    FPD\_RenderOptions opts,
    FS\_INT32 nClrMode
);
```

**Description**

Sets display mode code.

**Parameter**

---

opts	[In] The input page rendering options object.
------	---

---

nClrMode	[In] The new mode code.
----------	-------------------------

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 48

**FPDRenderOptionsSetForeColor****Syntax**

```
void FPDRenderOptionsSetForeColor (
    FPD\_RenderOptions opts,
    FS\_COLORREF clr
);
```

**Description**

Sets the foreground color for gray mode (default: black).

**Parameter**

---

opts	[In] The input page rendering options object.
------	---

---

clr	[In] Input the foreground color for gray mode (default: black).
-----	---

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 86

**FPDRenderOptionsSetOCCHandler****Syntax**

```
void FPDRenderOptionsSetOCCHandler (
    FPD\_RenderOptions opts,
    FPD\_OCContextHandler OCCHandler
);
```

**Description**

Sets optional content context handler.

**Parameter**

---

opts	[In] The input page rendering options object.
------	---

---

OCCHandler	[In] Input optional content context handler.
------------	--

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 134

**FPDRenderOptionsSetRenderFlag****Syntax**

```
void FPDRenderOptionsSetRenderFlag (
    FPD\_RenderOptions opts,
    FS\_DWORD dwFlag
);
```

**Description**

Sets render flags.

**Parameter**

---

opts	[In] The input page rendering options object.
------	---

---

dwFlag	[In] Input render flags.
--------	--------------------------

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 105

**FPDRenderOptionsTranslateColor****Syntax**

```
FS_ARGB FPDRenderOptionsTranslateColor (
    FPD\_RenderOptions opts,
    FS\_ARGB argb
);
```

**Description**

Translates a color.

**Parameter**

---

opts	[In] The input page rendering options object.
------	---

---

argb	[In] The input color.
------	-----------------------

---

**Return**

The translated color.

**Head file reference**

fpd\_renderTempl.h: 144

## FPD\_Rendition

**[Return from](#) [Used by](#)****Description**

A Rendition object. See [FPDRenditionNew](#) , [FPDRenditionNewFromDict](#) , [FPDRenditionDestroy](#) .

**Returned from**

[FPDRenditionNew](#)  
[FPDRenditionNewFromDict](#)

**Used by**

[FPDACTIONGetRendition](#)  
[FPDRenditionAddMediaPlayer](#)

[FPDRenditionAutoPlay](#)  
[FPDRenditionCountMediaPlayers](#)  
[FPDRenditionDestroy](#)  
[FPDRenditionEnableAutoPlay](#)  
[FPDRenditionEnableControlBarVisible](#)  
[FPDRenditionEnableFloatingWindowCloseButton](#)  
[FPDRenditionEnableFloatingWindowTitleBar](#)  
[FPDRenditionGetBackgroundColor](#)  
[FPDRenditionGetBackgroundOpacity](#)  
[FPDRenditionGetDuration](#)  
[FPDRenditionGetFitStyle](#)  
[FPDRenditionGetFloatingWindowOffscreen](#)  
[FPDRenditionGetFloatingWindowPosition](#)  
[FPDRenditionGetFloatingWindowRelativeType](#)  
[FPDRenditionGetFloatingWindowResizeType](#)  
[FPDRenditionGetFloatingWindowSize](#)  
[FPDRenditionGetFloatingWindowTitle](#)  
[FPDRenditionGetMediaBaseURL](#)  
[FPDRenditionGetMediaClipContentType](#)  
[FPDRenditionGetMediaClipFile](#)  
[FPDRenditionGetMediaClipName](#)  
[FPDRenditionGetMediaDescriptions](#)  
[FPDRenditionGetMediaPlayer](#)  
[FPDRenditionGetMonitor](#)  
[FPDRenditionGetPermission](#)  
[FPDRenditionGetRenditionName](#)  
[FPDRenditionGetVolume](#)  
[FPDRenditionGetWindowStatus](#)  
[FPDRenditionHasFloatingWindowCloseButton](#)  
[FPDRenditionHasFloatingWindowTitleBar](#)  
[FPDRenditionHasMediaClip](#)  
[FPDRenditionIsControlBarVisible](#)  
[FPDRenditionRemoveMediaPlayer](#)  
[FPDRenditionRepeatCount](#)  
[FPDRenditionSetBackgroundColor](#)  
[FPDRenditionSetBackgroundOpacity](#)  
[FPDRenditionSetDuration](#)  
[FPDRenditionSetFitStyle](#)  
[FPDRenditionSetFloatingWindowOffscreen](#)  
[FPDRenditionSetFloatingWindowPosition](#)  
[FPDRenditionSetFloatingWindowRelativeType](#)  
[FPDRenditionSetFloatingWindowResizeType](#)  
[FPDRenditionSetFloatingWindowSize](#)  
[FPDRenditionSetFloatingWindowTitle](#)  
[FPDRenditionSetMediaBaseURL](#)  
[FPDRenditionSetMediaClipContentType](#)  
[FPDRenditionSetMediaClipFile](#)  
[FPDRenditionSetMediaClipName](#)  
[FPDRenditionSetMediaDescriptions](#)  
[FPDRenditionSetMonitor](#)  
[FPDRenditionSetPermission](#)  
[FPDRenditionSetRenditionName](#)  
[FPDRenditionSetRepeatCount](#)  
[FPDRenditionSetVolume](#)  
[FPDRenditionSetWindowStatus](#)

## Functions

### Functions summary

#### [\*\*FPDRenditionAddMediaPlayer\*\*](#)

Adds a media player.

#### [\*\*FPDRenditionAutoPlay\*\*](#)

Checks whether the media should automatically play when activated.

#### [\*\*FPDRenditionCountMediaPlayers\*\*](#)

Gets the count of media players of specified type.

#### [\*\*FPDRenditionDestroy\*\*](#)

Destroys a rendition object.

#### [\*\*FPDRenditionEnableAutoPlay\*\*](#)

Sets the auto-play flag.

#### [\*\*FPDRenditionEnableControlBarVisible\*\*](#)

Sets the control bar visibility flag.

#### [\*\*FPDRenditionEnableFloatingWindowCloseButton\*\*](#)

Sets the visibility flag of close-button.

#### [\*\*FPDRenditionEnableFloatingWindowTitleBar\*\*](#)

Sets the title bar visibility flag.

#### [\*\*FPDRenditionGetBackgroundColor\*\*](#)

Gets the background color for the rectangle in which the media is being played.

#### [\*\*FPDRenditionGetBackgroundOpacity\*\*](#)

Gets the background opacity.

#### [\*\*FPDRenditionGetDuration\*\*](#)

Gets the intrinsic duration.

#### [\*\*FPDRenditionGetFitStyle\*\*](#)

Gets the fit style (manner) in which the player should treat a visual media type that does not exactly fit the rectangle in which it plays.

#### [\*\*FPDRenditionGetFloatingWindowOffscreen\*\*](#)

Gets what should occur if the floating window is positioned totally or partially off screen.

#### [\*\*FPDRenditionGetFloatingWindowPosition\*\*](#)

Gets the floating window position.

#### [\*\*FPDRenditionGetFloatingWindowRelativeType\*\*](#)

Gets the window type relative to which the floating window should be positioned.

#### [\*\*FPDRenditionGetFloatingWindowResizeType\*\*](#)

Checks whether the floating window may be resized by a user.

#### [\*\*FPDRenditionGetFloatingWindowSize\*\*](#)

Gets the floating window's width and height.

#### [\*\*FPDRenditionGetFloatingWindowTitle\*\*](#)

Gets the multi-language text array providing text to display on the floating window's title bar.

#### [\*\*FPDRenditionGetMediaBaseURL\*\*](#)

Gets an absolute URL to be used as the base URL in resolving any relative URLs found within the media data.

#### [\*\*FPDRenditionGetMediaClipContentType\*\*](#)

Gets the content type (MIME type) of the media data.

#### [\*\*FPDRenditionGetMediaClipFile\*\*](#)

Gets the file specification of the actual media data.

#### [\*\*FPDRenditionGetMediaClipName\*\*](#)

Gets the media clip name.

**FPDRenditionGetMediaDescriptions**

Gets the text descriptions array that provides alternate text descriptions for the media clip data in case it cannot be played.

**FPDRenditionGetMediaPlayer**

Gets a media player.

**FPDRenditionGetMonitor**

Gets the monitor specifier that specifies which monitor in a multi-monitor system a floating or full-screen window should appear on.

**FPDRenditionGetPermission**

Gets the media permission that indicates the circumstances under which it is acceptable to write a temporary file in order to play a media clip.

**FPDRenditionGetRenditionName**

Gets the rendition name.

**FPDRenditionGetVolume**

Gets the volume that specifies the desired volume level as a percentage of recorded volume level.

**FPDRenditionGetWindowStatus**

Gets the window status(type) that the media object should play in.

**FPDRenditionHasFloatingWindowCloseButton**

Checks whether the floating window should include user interface elements that allow a user to close a floating window.

**FPDRenditionHasFloatingWindowTitleBar**

Checks whether the floating window should have a title bar.

**FPDRenditionHasMediaClip**

Checks whether the rendition dictionary has the C entry.

**FPDRenditionIsControlBarVisible**

Checks to display a player-specific controller user interface (for example, play/pause/stop controls) when playing.

**FPDRenditionNew**

Creates an empty rendition object.

**FPDRenditionNewFromDict**

Creates a rendition object from a PDF dictionary.

**FPDRenditionRemoveMediaPlayer**

Removes a media player.

**FPDRenditionRepeatCount**

Gets the repeat count.

**FPDRenditionSetBackgroundColor**

Sets the background color.

**FPDRenditionSetBackgroundOpacity**

Sets the background opacity.

**FPDRenditionSetDuration**

Sets the intrinsic duration.

**FPDRenditionSetFitStyle**

Sets the fit style.

**FPDRenditionSetFloatingWindowOffscreen**

Sets the action Function.

**FPDRenditionSetFloatingWindowPosition**

Sets the floating window position.

**FPDRenditionSetFloatingWindowRelativeType**

Sets the window relative type.

**FPDRenditionSetFloatingWindowResizeType**

Sets the floating window resizing type.

**FPDRenditionSetFloatingWindowSize**

Sets the floating window's width and height.

**FPDRenditionSetFloatingWindowTitle**

Sets the multi-language text array to provide text for title bar.

**FPDRenditionSetMediaBaseURL**

Sets the base URL.

**FPDRenditionSetMediaClipContentType**

Sets the content type (MIME type) of the media data.

**FPDRenditionSetMediaClipFile**

Sets the file specification of the actual media data.

**FPDRenditionSetMediaClipName**

Sets the media clip name.

**FPDRenditionSetMediaDescriptions**

Sets the text descriptions array.

**FPDRenditionSetMonitor**

Sets the monitor specifier.

**FPDRenditionSetPermission**

Sets the media permission.

**FPDRenditionSetRenditionName**

Sets the rendition name.

**FPDRenditionSetRepeatCount**

Sets the repeat count.

**FPDRenditionSetVolume**

Sets the volume.

**FPDRenditionSetWindowStatus**

Sets the window status.

## Functions detail

### FPDRenditionAddMediaPlayer

#### Syntax

```
FS_INT32 FPDRenditionAddMediaPlayer (
    FPD_Rendition rendition,
    FPD_MediaPlayerType ePlayerType,
    FPD_MediaPlayer player
);
```

#### Description

Adds a media player.

#### Parameter

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

ePlayerType	[In] The input media player type.
-------------	-----------------------------------

---

---

player	[In] The input media player.
--------	------------------------------

---

**Return**

The current count of specified media player type.

**Head file reference**

fpd\_docTempl.h: 2819

**FPDRenditionAutoPlay****Syntax**

```
FS_BOOL FPDRenditionAutoPlay (
    FPD\_Rendition rendition
);
```

**Description**

Checks whether the media should automatically play when activated.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

[TRUE](#) for the media automatically playing when activated, otherwise not.

**Head file reference**

fpd\_docTempl.h: 2922

**FPDRenditionCountMediaPlayers****Syntax**

```
FS_INT32 FPDRenditionCountMediaPlayers (
    FPD\_Rendition rendition,
    FPD\_MediaPlayerType ePlayerType
);
```

**Description**

Gets the count of media players of specified type.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

ePlayerType	[In] The input media player type.
-------------	-----------------------------------

---

**Return**

The count of media player of specified type.

**Head file reference**

fpd\_docTempl.h: 2797

**FPDRenditionDestroy****Syntax**

```
void FPDRenditionDestroy (
    FPD\_Rendition rendition
);
```

**Description**

Destroys a rendition object.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2636

**FPDRenditionEnableAutoPlay****Syntax**

```
void FPDRenditionEnableAutoPlay (
    FPD\_Rendition rendition,
    FS\_BOOL bAuto,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the auto-play flag.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

bAuto	[In] The input auto-play flag.
-------	--------------------------------

---

eParamType	[In] The input media play param type.
------------	---------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2931

**FPDRenditionEnableControlBarVisible****Syntax**

```
void FPDRenditionEnableControlBarVisible (
    FPD\_Rendition rendition,
    FS\_BOOL bVisible,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the control bar visibility flag.

**Parameter**

rendition	[In] The input rendition object.
bVisible	[In] The input visibility flag of the control bar.
eParamType	[In] The input media play param type.

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2870

**FPDRenditionEnableFloatingWindowCloseButton****Syntax**

```
void FPDRenditionEnableFloatingWindowCloseButton (
    FPD\_Rendition rendition,
    FS\_BOOL bVisible,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the visibility flag of close-button.

**Parameter**

---

rendition	[In] The input rendition object.
bVisible	[In] The input visibility flag.
eParamType	[In] The input media play param type.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3155

**FPDRenditionEnableFloatingWindowTitleBar****Syntax**

```
void FPDRenditionEnableFloatingWindowTitleBar (
    FPD\_Rendition rendition,
    FS\_BOOL bVisible,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the title bar visibility flag.

**Parameter**


---

rendition	[In] The input rendition object.
bVisible	[In] The input visibility flag.
eParamType	[In] The input media play param type.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3135

**FPDRenditionGetBackgroundColor****Syntax**

```
FS_COLORREF FPDRenditionGetBackgroundColor (
    FPD\_Rendition rendition
);
```

**Description**

Gets the background color for the rectangle in which the media is being played.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

The background color for the rectangle in which the media is being played.

**Head file reference**

fpd\_docTempl.h: 2982

**FPDRenditionGetBackgroundOpacity****Syntax**

```
FS_FLOAT FPDRenditionGetBackgroundOpacity (
    FPD\_Rendition rendition
);
```

**Description**

Gets the background opacity.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

The background opacity.

**Head file reference**

fpd\_docTempl.h: 3002

**FPDRenditionGetDuration****Syntax**

```
FS_INT32 FPDRenditionGetDuration (
    FPD\_Rendition rendition
);
```

**Description**

Gets the intrinsic duration.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

The intrinsic duration.

**Head file reference**

fpd\_docTempl.h: 2902

**FPDRenditionGetFitStyle****Syntax**

```
FS_INT32 FPDRenditionGetFitStyle (
    FPD\_Rendition rendition
);
```

**Description**

Gets the fit style (manner) in which the player should treat a visual media type that does not exactly fit the rectangle in which it plays.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

The fit style (manner).

**Head file reference**

fpd\_docTempl.h: 2881

**FPDRenditionGetFloatingWindowOffscreen****Syntax**

```
FS_INT32 FPDRenditionGetFloatingWindowOffscreen (
    FPD\_Rendition rendition
);
```

**Description**

Gets what should occur if the floating window is positioned totally or partially off screen.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

0: nothing; 1: move or resize; 2: consider the object non-viable.

**Head file reference**

fpd\_docTempl.h: 3106

**FPDRenditionGetFloatingWindowPosition****Syntax**

```
FS_INT32 FPDRenditionGetFloatingWindowPosition (
    FPD\_Rendition rendition
);
```

**Description**

Gets the floating window position.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

0: upper left; 1: upper center; 2: upper right; 3: center left; 4: center; 5: center right; 6: lower left; 7: lower center; 8: lower right.

**Head file reference**

fpd\_docTempl.h: 3086

**FPDRenditionGetFloatingWindowRelativeType****Syntax**

```
FS_INT32 FPDRenditionGetFloatingWindowRelativeType (
    FPD\_Rendition rendition
);
```

**Description**

Gets the window type relative to which the floating window should be positioned.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

0: document window; 1: application window; 2: full virtual desktop; 3: monitor screen.

**Head file reference**

fpd\_docTempl.h: 3066

**FPDRenditionGetFloatingWindowResizeType**

**Syntax**

```
FS_INT32 FPDREnditionGetFloatingWindowResizeType (
    FPD Rendition rendition
);
```

**Description**

Checks whether the floating window may be resized by a user.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

0: may not be resized; 1: resize with aspect ratio; 2: resize anyway.

**Head file reference**

fpd\_docTempl.h: 3166

**FPDRenditionGetFloatingWindowSize****Syntax**

```
FS_BOOL FPDREnditionGetFloatingWindowSize (
    FPD Rendition rendition,
    FS_INT32* iWidth,
    FS_INT32* iHeight
);
```

**Description**

Gets the floating window's width and height.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

iWidth	[Out] It receives the floating window's width.
--------	--

---

iHeight	[Out] It receives the floating window's height.
---------	---

---

**Return**

[TRUE](#) : returned size is available; [FALSE](#) : not available.

**Head file reference**

fpd\_docTempl.h: 3043

**FPDRenditionGetFloatingWindowTitle**

**Syntax**

```
FS_BOOL FPDREnditionGetFloatingWindowTitle (
    FPD\_Rendition rendition,
    FS\_WideStringArray* outTitleArray
);
```

**Description**

Gets the multi-language text array providing text to display on the floating window's title bar.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

outTitleArray	[Out] It receives the multi-language text array.
---------------	--

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_docTempl.h: 3186

**FPDRenditionGetMediaBaseURL****Syntax**

```
void FPDREnditionGetMediaBaseURL (
    FPD\_Rendition rendition,
    FS\_ByteString* outURL
);
```

**Description**

Gets an absolute URL to be used as the base URL in resolving any relative URLs found within the media data.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

outURL	[Out] It receives the absolute URL.
--------	-------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2776

## FPDRenditionGetMediaClipContentType

### Syntax

```
void FPDRenditionGetMediaClipContentType (
    FPD\_Rendition rendition,
    FS\_ByteString* outContentType
);
```

### Description

Gets the content type (MIME type) of the media data.

### Parameter

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

outContentType	[Out] The content type (MIME type) of the media data.
----------------	---

---

### Return

void

#### Head file reference

fpd\_docTempl.h: 2715

## FPDRenditionGetMediaClipFile

### Syntax

```
void FPDRenditionGetMediaClipFile (
    FPD\_Rendition rendition,
    FPD\_FileSpec* outFileSpec
);
```

### Description

Gets the file specification of the actual media data.

### Parameter

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

outFileSpec	[Out] The file specification of the actual media data.
-------------	--

---

### Return

void

#### Head file reference

fpd\_docTempl.h: 2694

## FPDRenditionGetMediaClipName

### Syntax

```
void FPDRenditionGetMediaClipName (
    FPD\_Rendition rendition,
    FS\_WideString* outName
);
```

### Description

Gets the media clip name.

### Parameter

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

outName	[Out] The media clip name.
---------	----------------------------

---

### Return

void

### Head file reference

fpd\_docTempl.h: 2674

## FPDRenditionGetMediaDescriptions

### Syntax

```
FS_BOOL FPDRenditionGetMediaDescriptions (
    FPD\_Rendition rendition,
    FS\_WideStringArray* outDescArray
);
```

### Description

Gets the text descriptions array that provides alternate text descriptions for the media clip data in case it cannot be played.

### Parameter

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

outDescArray	[Out] It receives the text descriptions array.
--------------	--

---

### Return

Non-zero means success, otherwise failure.

### Head file reference

fpd\_docTempl.h: 2755

**FPDRenditionGetMediaPlayer****Syntax**

```
void FPDRenditionGetMediaPlayer (
    FPD\_Rendition rendition,
    FPD\_MediaPlayerType ePlayerType,
    FS\_INT32 index,
    FPD\_MediaPlayer* outMediaPlayer
);
```

**Description**

Gets a media player.

**Parameter**

rendition	[In] The input rendition object.
-----------	----------------------------------

ePlayerType	[In] The input media player type.
-------------	-----------------------------------

index	[In] The zero-based index in the specified media player type.
-------	---

outMediaPlayer	[Out] A media player.
----------------	-----------------------

**Return**

void

**Head file reference**

`fpd_docTempl.h: 2807`

**FPDRenditionGetMonitor****Syntax**

```
FS\_INT32 FPDRenditionGetMonitor (
    FPD\_Rendition rendition
);
```

**Description**

Gets the monitor specifier that specifies which monitor in a multi-monitor system a floating or full-screen window should appear on.

**Parameter**

rendition	[In] The input rendition object.
-----------	----------------------------------

**Return**

The monitor specifier

**Head file reference**

fpd\_docTempl.h: 3022

**FPDRenditionGetPermission****Syntax**

```
FPD_MediaPermission FPDRenditionGetPermission (
    FPD\_Rendition rendition
);
```

**Description**

Gets the media permission that indicates the circumstances under which it is acceptable to write a temporary file in order to play a media clip.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

The media permission

**Head file reference**

fpd\_docTempl.h: 2735

**FPDRenditionGetRenditionName****Syntax**

```
void FPDRenditionGetRenditionName (
    FPD\_Rendition rendition,
    FS\_WideString* outName
);
```

**Description**

Gets the rendition name.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

outName	[Out] The rendition name.
---------	---------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2654

## FPDRenditionGetVolume

### Syntax

```
FS_INT32 FPDRenditionGetVolume (
    FPD Rendition rendition
);
```

### Description

Gets the volume that specifies the desired volume level as a percentage of recorded volume level.

### Parameter

---

rendition	[In] The input rendition object.
-----------	----------------------------------

### Return

The volume

### Head file reference

fpd\_docTempl.h: 2841

## FPDRenditionGetWindowStatus

### Syntax

```
FS_INT32 FPDRenditionGetWindowStatus (
    FPD Rendition rendition
);
```

### Description

Gets the window status(type) that the media object should play in.

### Parameter

---

rendition	[In] The input rendition object.
-----------	----------------------------------

### Return

0: floating window; 1: full screen; 2: hidden window; 3: child window.

### Head file reference

fpd\_docTempl.h: 2962

## FPDRenditionHasFloatingWindowCloseButton

### Syntax

```
FS_BOOL FPDRenditionHasFloatingWindowCloseButton (
    FPD Rendition rendition
```

);

**Description**

Checks whether the floating window should include user interface elements that allow a user to close a floating window.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

[TRUE](#) for the floating window including user interface elements, otherwise not.

**Head file reference**

fpd\_docTempl.h: 3146

**FPDRenditionHasFloatingWindowTitleBar****Syntax**

```
FS_BOOL FPDRenditionHasFloatingWindowTitleBar (
    FPD\_Rendition rendition
);
```

**Description**

Checks whether the floating window should have a title bar.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

[TRUE](#) for the floating window having a title bar.

**Head file reference**

fpd\_docTempl.h: 3126

**FPDRenditionHasMediaClip****Syntax**

```
FS_BOOL FPDRenditionHasMediaClip (
    FPD\_Rendition rendition
);
```

**Description**

Checks whether the rendition dictionary has the C entry.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

TRUE for the rendition dictionary having the C entry, otherwise not.

**Head file reference**

fpd\_docTempl.h: 2645

**FPDRenditionIsControlBarVisible****Syntax**

```
FS_BOOL FPDRenditionIsControlBarVisible (
    FPD_Rendition rendition
);
```

**Description**

Checks to display a player-specific controller user interface (for example, play/pause/stop controls) when playing.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

**Return**

TRUE for being visible, otherwise not.

**Head file reference**

fpd\_docTempl.h: 2861

**FPDRenditionNew****Syntax**

```
FPD_Rendition FPDRenditionNew (void );
```

**Description**

Creates an empty rendition object.

**Return**

An empty rendition object.

**Head file reference**

fpd\_docTempl.h: 2618

## FPDRenditionNewFromDict

### Syntax

```
FPD_Rendition FPDRenditionNewFromDict (  
    FPD\_Object dict  
);
```

### Description

Creates a rendition object from a PDF dictionary.

### Parameter

---

dict	[In] The input PDF dictionary.
------	--------------------------------

---

### Return

A rendition object.

### Head file reference

fpd\_docTempl.h: 2627

## FPDRenditionRemoveMediaPlayer

### Syntax

```
void FPDRenditionRemoveMediaPlayer (  
    FPD\_Rendition rendition,  
    FPD\_MediaPlayerType ePlayerType,  
    FPD\_MediaPlayer player  
);
```

### Description

Removes a media player.

### Parameter

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

ePlayerType	[In] The input media player type.
-------------	-----------------------------------

---

player	[In] The media player to be removed.
--------	--------------------------------------

---

### Return

void

### Head file reference

fpd\_docTempl.h: 2830

## FPDRenditionRepeatCount

### Syntax

```
FS_INT32 FPDRenditionRepeatCount (
    FPD\_Rendition rendition
);
```

### Description

Gets the repeat count.

### Parameter

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

### Return

The repeat count.

### Head file reference

fpd\_docTempl.h: 2942

## FPDRenditionSetBackgroundColor

### Syntax

```
void FPDRenditionSetBackgroundColor (
    FPD\_Rendition rendition,
    FS\_COLORREF color,
    FPD\_MediaPlayParamType eParamType
);
```

### Description

Sets the background color.

### Parameter

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

color	[In] The input background color.
-------	----------------------------------

---

eParamType	[In] The input media play param type.
------------	---------------------------------------

---

### Return

void

### Head file reference

fpd\_docTempl.h: 2991

**FPDRenditionSetBackgroundOpacity****Syntax**

```
void FPDRenditionSetBackgroundOpacity (
    FPD\_Rendition rendition,
    FS\_FLOAT iOpacity,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the background opacity.

**Parameter**

rendition	[In] The input rendition object.
-----------	----------------------------------

iOpacity	[In] The input background opacity.
----------	------------------------------------

eParamType	[In] The input media play param type.
------------	---------------------------------------

**Return**

void

**Head file reference**

[fpd\\_docTempl.h](#): 3011

**FPDRenditionSetDuration****Syntax**

```
void FPDRenditionSetDuration (
    FPD\_Rendition rendition,
    FS\_INT32 iDuration,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the intrinsic duration.

**Parameter**

rendition	[In] The input rendition object.
-----------	----------------------------------

iDuration	[In] The input intrinsic duration.
-----------	------------------------------------

eParamType	[In] The input media play param type.
------------	---------------------------------------

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2911

**FPDRenditionSetFitStyle****Syntax**

```
void FPDRenditionSetFitStyle (
    FPD\_Rendition rendition,
    FS\_INT32 iFitStyle,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the fit style.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

iFitStyle	[In] The input fit style.
-----------	---------------------------

---

eParamType	[In] The input media play param type.
------------	---------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2891

**FPDRenditionSetFloatingWindowOffscreen****Syntax**

```
void FPDRenditionSetFloatingWindowOffscreen (
    FPD\_Rendition rendition,
    FS\_INT32 iOffscreen,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the action Function.

**Parameter**

---

rendition	[In] The input rendition object.
iOffscreen	[In] The input action Function.0: nothing; 1: move or resize; 2: consider the object non-viable.
eParamType	[In] The input media play param type.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3115

**FPDRenditionSetFloatingWindowPosition****Syntax**

```
void FPDRenditionSetFloatingWindowPosition (
    FPD\_Rendition rendition,
    FS\_INT32 iPosition,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the floating window position.

**Parameter**


---

rendition	[In] The input rendition object.
iPosition	[In] The input floating window position. 0: upper left; 1: upper center; 2: upper right; 3: center left; 4: center; 5: center right; 6: lower left; 7: lower center; 8: lower right.
eParamType	[In] The input media play param type.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3095

**FPDRenditionSetFloatingWindowRelativeType****Syntax**

```
void FPDRenditionSetFloatingWindowRelativeType (
```

---

```
FPD_Rendition rendition,
FS_INT32 iType,
FPD_MediaPlayParamType eParamType
);
```

**Description**

Sets the window relative type.

**Parameter**


---

rendition	[In] The input rendition object.
iType	[In] The input window relative type. 0: document window; 1: application window; 2: full virtual desktop; 3: monitor screen.
eParamType	[In] The input media play param type.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3075

**FPDRenditionSetFloatingWindowResizeType****Syntax**

```
void FPDRenditionSetFloatingWindowResizeType (
    FPD_Rendition rendition,
    FS_INT32 iType,
    FPD_MediaPlayParamType eParamType
);
```

**Description**

Sets the floating window resizing type.

**Parameter**


---

rendition	[In] The input rendition object.
iType	[In] The input floating window resizing type. 0: may not be resized; 1: resize with aspect ratio; 2: resize anyway.
eParamType	[In] The input media play param type.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3175

**FPDRenditionSetFloatingWindowSize****Syntax**

```
void FPDRenditionSetFloatingWindowSize (
    FPD\_Rendition rendition,
    FS\_INT32 iWidth,
    FS\_INT32 iHeight,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the floating window's width and height.

**Parameter**

rendition	[In] The input rendition object.
-----------	----------------------------------

iWidth	[In] The width of the floating window.
--------	--

iHeight	[In] The height of the floating window.
---------	---

eParamType	[In] The input media play param type.
------------	---------------------------------------

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3054

**FPDRenditionSetFloatingWindowTitle****Syntax**

```
void FPDRenditionSetFloatingWindowTitle (
    FPD\_Rendition rendition,
    FS\_WideStringArray titleArray,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the multi-language text array to provide text for title bar.

**Parameter**

---

rendition	[In] The input rendition object.
titleArray	[In] The input multi-language text array.
eParamType	[In] The input media play param type.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3196

**FPDRenditionSetMediaBaseURL****Syntax**

```
void FPDRenditionSetMediaBaseURL (
    FPD\_Rendition rendition,
    FS\_LPCSTR szBaseUrl,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the base URL.

**Parameter**


---

rendition	[In] The input rendition object.
szBaseUrl	[In] The input base URL.
eParamType	[In] The input media play param type.

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2786

**FPDRenditionSetMediaClipContentType****Syntax**

```
void FPDRenditionSetMediaClipContentType (
    FPD\_Rendition rendition,
    FS\_LPCSTR szContentType
);
```

**Description**

Sets the content type (MIME type) of the media data.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

szContentType	[In] The input content type.
---------------	------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2725

**FPDRenditionSetMediaClipFile****Syntax**

```
void FPDRenditionSetMediaClipFile (
    FPD\_Rendition rendition,
    FPD\_Document doc,
    FPD\_FileSpec file
);
```

**Description**

Sets the file specification of the actual media data.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

doc	[In] The input PDF document.
-----	------------------------------

---

file	[In] The input file specification object.
------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2704

**FPDRenditionSetMediaClipName**

**Syntax**

```
void FPDREnditionSetMediaClipName (
    FPD\_Rendition rendition,
    FS\_LPCWSTR wszName
);
```

**Description**

Sets the media clip name.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

wszName	[In] The input media clip name.
---------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2684

**FPDREnditionSetMediaDescriptions****Syntax**

```
void FPDREnditionSetMediaDescriptions (
    FPD\_Rendition rendition,
    const FS\_WideStringArray descArray
);
```

**Description**

Sets the text descriptions array.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

descArray	[In] The input text descriptions array.
-----------	---

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2766

**FPDREnditionSetMonitor**

**Syntax**

```
void FPDREnditionSetMonitor (
    FPD\_Rendition rendition,
    FS\_INT32 iMonitor,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the monitor specifier.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

iMonitor	[In] The input monitor specifier.
----------	-----------------------------------

---

eParamType	[In] The input media play param type.
------------	---------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 3032

**FPDRenditionSetPermission****Syntax**

```
void FPDREnditionSetPermission (
    FPD\_Rendition rendition,
    FPD\_MediaPermission ePermission
);
```

**Description**

Sets the media permission.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

ePermission	[In] The input media permission.
-------------	----------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2745

## FPDRenditionSetRenditionName

### Syntax

```
void FPDRenditionSetRenditionName (
```

[FPD\\_Rendition](#) rendition,  
[FS\\_LPCWSTR](#) wszName  
);

### Description

Sets the rendition name.

### Parameter

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

wszName	[In] The input rendition name.
---------	--------------------------------

---

### Return

void

### Head file reference

fpd\_docTempl.h: 2664

## FPDRenditionSetRepeatCount

### Syntax

```
void FPDRenditionSetRepeatCount (
```

[FPD\\_Rendition](#) rendition,  
[FS\\_INT32](#) iCount,  
[FPD\\_MediaPlayParamType](#) eParamType  
);

### Description

Sets the repeat count.

### Parameter

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

iCount	[In] The input repeat count.
--------	------------------------------

---

eParamType	[In] The input media play param type.
------------	---------------------------------------

---

### Return



void

**Head file reference**

fpd\_docTempl.h: 2951

**FPDRenditionSetVolumn****Syntax**

```
void FPDRenditionSetVolumn (
    FPD\_Rendition rendition,
    FS\_INT32 iVolumn,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the volume.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

iVolumn	[In] The input volume.
---------	------------------------

---

eParamType	[In] The media play param type.
------------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2850

**FPDRenditionSetWindowStatus****Syntax**

```
void FPDRenditionSetWindowStatus (
    FPD\_Rendition rendition,
    FS\_INT32 iStatus,
    FPD\_MediaPlayParamType eParamType
);
```

**Description**

Sets the window status.

**Parameter**

---

rendition	[In] The input rendition object.
-----------	----------------------------------

---

---

iStatus	[In] The input window status.0: floating window; 1: full screen; 2: hidden window; 3: child window.
---------	--

---

eParamType	[In] The input media play param type.
------------	---------------------------------------

---

**Return**

void

**Head file reference**

fpd\_docTempl.h: 2971

## FPD\_ShadingObject

### Description

A FPD\_PathObject is a graphic object (a subclass of FPD\_PageObject) representing a smooth shading in a page description. FPD\_PathObject describes geometric shapes whose color are an arbitrary function of position within the shape. (A shading can also be treated as a color when painting other graphics objects; it is not considered to be a separate graphics object in that case.)

### Functions

#### Functions summary

**FPDShadingObjectCalcBoundingBox**

Calculates the bounding box.

**FPDShadingObjectDestroy**

Destroys the PDF shading object.

**FPDShadingObjectGetPage**

Gets the page.

**FPDShadingObjectGetShadingPattern**

Gets the shading pattern.

**FPDShadingObjectGetTransformMatrix**

Gets the transformation matrix.

**FPDShadingObjectNew**

Creates a new empty PDF shading object.

**FPDShadingObjectSetPage**

Sets the page.

**FPDShadingObjectSetShadingPattern**

Sets the shading pattern.

**FPDShadingObjectSetTransformMatrix**

Sets the transformation matrix.

**FPDShadingObjectTransform**

Transforms the path object.

### Functions detail

**FPDShadingObjectCalcBoundingBox**

**Syntax**

```
void FPDShadingObjectCalcBoundingBox (
    FPD PageObject objShading
);
```

**Description**

Calculates the bounding box.

**Parameter**

---

objShading	[In] The input PDF shading object.
------------	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1725

**FPDShadingObjectDestroy****Syntax**

```
void FPDShadingObjectDestroy (
    FPD PageObject objShading
);
```

**Description**

Destroys the PDF shading object.

**Parameter**

---

objShading	[In] The input PDF shading object.
------------	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1706

**FPDShadingObjectGetPage****Syntax**

```
FPD_Page FPDShadingObjectGetPage (
    FPD PageObject objShading
);
```

**Description**

Gets the page.

**Parameter**

---

objShading	[In] The input PDF shading object.
------------	------------------------------------

---

**Return**

The page.

**Head file reference**

fpd\_pageobjTempl.h: 1754

**FPDShadingObjectGetShadingPattern****Syntax**

```
FPD_ShadingPattern FPDShadingObjectGetShadingPattern (
    FPD\_PageObject objShading
);
```

**Description**

Gets the shading pattern.

**Parameter**

---

objShading	[In] The input PDF shading object.
------------	------------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1773

**FPDShadingObjectGetTransformMatrix****Syntax**

```
void FPDShadingObjectGetTransformMatrix (
    FPD\_PageObject objShading,
    FS\_AffineMatrix* outmatrix
);
```

**Description**

Gets the transformation matrix.

**Parameter**

---

objShading	[In] The input PDF shading object.
------------	------------------------------------

---

---

outmatrix	[Out] It receives the transformation matrix.
-----------	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1734

**FPDShadingObjectNew****Syntax**

FPD\_PageObject FPDShadingObjectNew (void );

**Description**

Creates a new empty PDF shading object.

**Return**

A new empty PDF shading object

**Head file reference**

fpd\_pageobjTempl.h: 1697

**FPDShadingObjectSetPage****Syntax**

```
void FPDShadingObjectSetPage (
    FPD\_PageObject objShading,
    FPD\_Page page
);
```

**Description**

Sets the page.

**Parameter**

---

objShading	[In] The input PDF shading object.
------------	------------------------------------

---

---

page	[In] The input page.
------	----------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1763

## FPDShadingObjectSetShadingPattern

### Syntax

```
void FPDShadingObjectSetShadingPattern (
    FPD\_PageObject objShading,
    FPD\_ShadingPattern pattern
);
```

### Description

Sets the shading pattern.

### Parameter

---

objShading	[In] The input PDF shading object.
------------	------------------------------------

---

pattern	[In] The input shading pattern.
---------	---------------------------------

---

### Return

void

### Head file reference

fpd\_pageobjTempl.h: 1782

## FPDShadingObjectSetTransformMatrix

### Syntax

```
void FPDShadingObjectSetTransformMatrix (
    FPD\_PageObject objShading,
    const FS\_AffineMatrix\* matrix
);
```

### Description

Sets the transformation matrix.

### Parameter

---

objShading	[In] The input PDF shading object.
------------	------------------------------------

---

matrix	[In] The transformation matrix.
--------	---------------------------------

---

### Return

void

### Head file reference

fpd\_pageobjTempl.h: 1744

## FPDShadingObjectTransform

### Syntax

```
void FPDShadingObjectTransform (
    FPD\_PageObject objShading,
    FS\_AffineMatrix matrix
);
```

### Description

Transforms the path object.

### Parameter

---

objShading	[In] The input PDF shading object.
------------	------------------------------------

---

matrix	[In] The transformation matrix.
--------	---------------------------------

---

### Return

void

### Head file reference

`fpd_pageobjTempl.h: 1715`

## FPD\_ShadingPattern

[Return from Used by](#)

### Description

PDF shading pattern. see *CHAPTER 4.6.3 in PDF reference* . Shading patterns provide a smooth transition between colors across an area to be painted, dependent of the resolution of any particular output device and without specifying the number of steps in the color transition. See [FPDShadingPatternNew](#) , [FPDShadingPatternDestroy](#) .

### Returned from

[FPDShadingObjectGetShadingPattern](#)  
[FPDShadingPatternNew](#)

### Used by

[FPDShadingObjectSetShadingPattern](#)  
[FPDShadingPatternDestroy](#)  
[FPDShadingPatternGetColorSpace](#)  
[FPDShadingPatternGetFunc](#)  
[FPDShadingPatternGetFuncsCount](#)  
[FPDShadingPatternGetPatternMatrix](#)  
[FPDShadingPatternGetPatternObj](#)  
[FPDShadingPatternGetPatternType](#)

---

[\*\*FPDShadingPatternGetPDDoc\*\*](#)  
[\*\*FPDShadingPatternGetShadingObject\*\*](#)  
[\*\*FPDShadingPatternLoad\*\*](#)  
[\*\*FPDShadingPatternReLoad\*\*](#)  
[\*\*FPDShadingPatternSetColorSpace\*\*](#)

## Functions

### Functions summary

[\*\*FPDShadingPatternDestroy\*\*](#)

Destroys the shading pattern object.

[\*\*FPDShadingPatternGetColorSpace\*\*](#)

Gets the color space.

[\*\*FPDShadingPatternGetFunc\*\*](#)

Gets the function pointer.

[\*\*FPDShadingPatternGetFuncsCount\*\*](#)

Gets the count of functions.

[\*\*FPDShadingPatternGetPatternMatrix\*\*](#)

Gets matrix from pattern to parent stream.

[\*\*FPDShadingPatternGetPatternObj\*\*](#)

Gets dictionary for shading, stream for tiling.

[\*\*FPDShadingPatternGetPatternType\*\*](#)

Gets the pattern type

[\*\*FPDShadingPatternGetPDDoc\*\*](#)

Gets the PDF document.

[\*\*FPDShadingPatternGetShadingObject\*\*](#)

Gets the PDF object of shading..

[\*\*FPDShadingPatternLoad\*\*](#)

Loads all following data.

[\*\*FPDShadingPatternNew\*\*](#)

Creates a new empty PDF shading pattern object.

[\*\*FPDShadingPatternReLoad\*\*](#)

Reloads shading data after shading dictionary changed .

[\*\*FPDShadingPatternSetColorSpace\*\*](#)

Sets the color space.

### Functions detail

#### FPDShadingPatternDestroy

##### Syntax

```
void FPDShadingPatternDestroy (
    FPD_ShadingPattern pattern
);
```

##### Description

Destroys the shading pattern object.

##### Parameter

---

pattern	[In] The shading pattern object to be destroyed.
---------	--

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 1766

**FPDShadingPatternGetColorSpace****Syntax**

```
FPD_ColorSpace FPDShadingPatternGetColorSpace (
    FPD\_ShadingPattern pattern
);
```

**Description**

Gets the color space.

**Parameter**

---

pattern	[In] The input PDF shading pattern object.
---------	--

---

**Return**

The color space.

**Head file reference**

fpd\_resourceTempl.h: 1829

**FPDShadingPatternGetFunc****Syntax**

```
FPD_Function FPDShadingPatternGetFunc (
    FPD\_ShadingPattern pattern,
    FS\_INT32 index
);
```

**Description**

Gets the function pointer.

**Parameter**

---

pattern	[In] The input PDF shading pattern object.
---------	--

---

---

index	[In] Function pointer index.
-------	------------------------------

---

**Return**

The function pointer.

**Head file reference**

fpd\_resourceTempl.h: 1838

**FPDShadingPatternGetFuncsCount****Syntax**

```
FS_INT32 FPDShadingPatternGetFuncsCount (
    FPD\_ShadingPattern pattern
);
```

**Description**

Gets the count of functions.

**Parameter**

---

pattern	[In] The input PDF shading pattern object.
---------	--

---

**Return**

The count of functions.

**Head file reference**

fpd\_resourceTempl.h: 1848

**FPDShadingPatternGetPatternMatrix****Syntax**

```
FS_AffineMatrix FPDShadingPatternGetPatternMatrix (
    FPD\_ShadingPattern pattern
);
```

**Description**

Gets matrix from pattern to parent stream.

**Parameter**

---

pattern	[In] The input PDF shading pattern object.
---------	--

---

**Return**

The matrix from pattern to parent stream.

**Head file reference**

fpd\_resourceTempl.h: 1793

**FPDShadingPatternGetPatternObj**

**Syntax**

```
FPD_Object FPDShadingPatternGetPatternObj (
    FPD\_ShadingPattern pattern
);
```

**Description**

Gets dictionary for shading, stream for tiling.

**Parameter**

---

pattern	[In] The input PDF shading pattern object.
---------	--

---

**Return**

A dictionary for shading, stream for tiling.

**Head file reference**

fpd\_resourceTempl.h: 1775

**FPDShadingPatternGetPatternType****Syntax**

```
FS_INT32 FPDShadingPatternGetPatternType (
    FPD\_ShadingPattern pattern
);
```

**Description**

Gets the pattern type

**Parameter**

---

pattern	[In] The input PDF shading pattern object.
---------	--

---

**Return**

The pattern type

**Head file reference**

fpd\_resourceTempl.h: 1784

**FPDShadingPatternGetPDDoc****Syntax**

```
FPD_Document FPDShadingPatternGetPDDoc (
    FPD\_ShadingPattern pattern
);
```

**Description**

Gets the PDF document.

#### Parameter

---

pattern	[In] The input PDF shading pattern object.
---------	--

---

#### Return

The PDF document.

#### Head file reference

fpd\_resourceTempl.h: 1802

## FPDShadingPatternGetShadingObject

#### Syntax

```
FPD_Object FPDShadingPatternGetShadingObject (
    FPD\_ShadingPattern pattern
);
```

#### Description

Gets the PDF object of shading..

#### Parameter

---

pattern	[In] The input PDF shading pattern object.
---------	--

---

#### Return

The PDF object of shading..

#### Head file reference

fpd\_resourceTempl.h: 1867

#### Since

[SDK\\_LATEEST\\_VERSION > 2.1.0.4](#)

## FPDShadingPatternLoad

#### Syntax

```
FS_BOOL FPDShadingPatternLoad (
    FPD\_ShadingPattern pattern
);
```

#### Description

Loads all following data.

#### Parameter

---

pattern	[In] The input PDF shading pattern object.
---------	--

---

**Return**

[TRUE](#) if the data is loaded, otherwise [FALSE](#) .

**Head file reference**

fpd\_resourceTempl.h: 1811

**FPDShadingPatternNew****Syntax**

```
FPD_ShadingPattern FPDShadingPatternNew (
    FPD\_Document doc,
    FPD\_Object obj,
    FS\_BOOL bShading
);
```

**Description**

Creates a new empty PDF shading pattern object.

**Parameter**


---

doc	[In] The PDF document.
-----	------------------------

---

obj	[In] The PDF object of pattern.
-----	---------------------------------

---

bShading	[In] Whether it's just a shading object.
----------	--

---

**Return**

A new shading pattern object.

**Head file reference**

fpd\_resourceTempl.h: 1755

**FPDShadingPatternReLoad****Syntax**

```
FS_BOOL FPDShadingPatternReLoad (
    FPD\_ShadingPattern pattern
);
```

**Description**

Reloads shading data after shading dictionary changed .

**Parameter**

---

pattern	[In] The input PDF shading pattern object.
---------	--

---

**Return**

[TRUE](#) if the data is reloaded, otherwise [FALSE](#).

**Head file reference**

fpd\_resourceTempl.h: 1820

**FPDShadingPatternSetColorSpace****Syntax**

```
void FPDShadingPatternSetColorSpace (
    FPD\_ShadingPattern pattern,
    FPD\_ColorSpace colorSpace
);
```

**Description**

Sets the color space.

**Parameter**


---

pattern	[In] The input PDF shading pattern object.
---------	--

---

colorSpace	[In] The input color space to be set.
------------	---------------------------------------

---

**Return**

void.

**Head file reference**

fpd\_resourceTempl.h: 1857

**FPD\_Stream**[\*\*Return from Used by\*\*](#)**Description**

A FPD\_Stream is a sequence of characters that can be read a portion at a time. Streams are used for objects with large amounts of data, such as images, page content, or private data a plug-in creates. A stream consists of these elements, which are listed in their relative order in the stream object, starting at the beginning. See Section 3.2.7 in the PDF Reference for a description of the stream object.

**Returned from**

[FPDDocLoadFontFile](#)  
[FPDFFontGetFontFile](#)

[\*\*FPDStreamAccNew\*\*](#)  
[\*\*FPDStreamGetStreamFilter\*\*](#)

**Used by**

[\*\*FPDStreamAccDestroy\*\*](#)  
[\*\*FPDStreamAccDetachData\*\*](#)  
[\*\*FPDStreamAccGetData\*\*](#)  
[\*\*FPDStreamAccGetDict\*\*](#)  
[\*\*FPDStreamAccGetImageDecoder\*\*](#)  
[\*\*FPDStreamAccGetImageParam\*\*](#)  
[\*\*FPDStreamAccGetSize\*\*](#)  
[\*\*FPDStreamAccGetStream\*\*](#)  
[\*\*FPDStreamFilterDestroy\*\*](#)  
[\*\*FPDStreamFilterGetSrcPos\*\*](#)  
[\*\*FPDStreamFilterGetStream\*\*](#)  
[\*\*FPDStreamFilterReadBlock\*\*](#)

## Functions

### Functions summary

[\*\*FPDStreamGetDict\*\*](#)

Gets the dictionary of the stream object.

[\*\*FPDStreamGetRawSize\*\*](#)

Gets the raw data size in bytes.

[\*\*FPDStreamGetStreamFilter\*\*](#)

Creates a stream-based data filter from the PDF stream. The filter can output either raw data (decrypted) or decoded data. JBIG2 and JPEG2000 decoding not supported. Caller must destroy the created filter.

[\*\*FPDStreamIdentical\*\*](#)

Compares value with another object.

[\*\*FPDStreamInitStream\*\*](#)

Initializes a stream with data and dictionary. If no dictionary specified, the old dictionary is retained.

[\*\*FPDStreamIsMemoryBased\*\*](#)

Test whether the stream data is memory-based.

[\*\*FPDStreamNew\*\*](#)

Creates a new empty stream object.

[\*\*FPDStreamReadRawData\*\*](#)

Reads raw data.

[\*\*FPDStreamSetData\*\*](#)

Sets stream data. If *pData* is **NULL**, just allocate stream buffer. The data can be uncompressed or compressed. If it's uncompressed, then previous filter info will be dropped (if any). If it's compressed, the caller should also maintain the filter information in the dictionary.

### Functions detail

**FPDStreamGetDict**

**Syntax**

FPD\_Object FPDStreamGetDict (

```
FPD_Object objStream  
);
```

**Description**

Gets the dictionary of the stream object.

**Parameter**

---

objStream	[In] The input stream object.
-----------	-------------------------------

---

**Return**

The dictionary of the stream object.

**Head file reference**

fpd\_objsTempl.h: 1209

**FPDStreamGetRawSize****Syntax**

```
FS_DWORD FPDStreamGetRawSize (  
    FPD_Object objStream  
);
```

**Description**

Gets the raw data size in bytes.

**Parameter**

---

objStream	[In] The input stream object.
-----------	-------------------------------

---

**Return**

The raw data size in bytes.

**Head file reference**

fpd\_objsTempl.h: 1257

**FPDStreamGetStreamFilter****Syntax**

```
FPD_StreamFilter FPDStreamGetStreamFilter (  
    FPD_Object objStream,  
    FS_BOOL bRaw  
);
```

**Description**

Creates a stream-based data filter from the PDF stream. The filter can output either raw data (decrypted) or decoded data. JBIG2 and JPEG2000 decoding not supported. Caller must destroy the created filter.

#### Parameter

---

objStream	[In] The input stream object.
bRaw	[In] Whether the stream filter will do decoding.

---

#### Return

A stream filter object.

#### Head file reference

fpd\_objsTempl.h: 1244

### FPDStreamIdentical

#### Syntax

```
FS_BOOL FPDStreamIdentical (
    FPD Object objStream,
    FPD Object other_stm
);
```

#### Description

Compares value with another object.

#### Parameter

---

objStream	[In] The input stream object.
other_stm	[In] The another stream object.

---

#### Return

Non-zero means identical, otherwise not identical.

#### Head file reference

fpd\_objsTempl.h: 1234

### FPDStreamInitStream

#### Syntax

```
void FPDStreamInitStream (void );
```

#### Description

Initializes a stream with data and dictionary. If no dictionary specified, the old dictionary is retained.

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1200

**FPDStreamIsMemoryBased****Syntax**

```
FS_BOOL FPDStreamIsMemoryBased (
    FPD Object objStream
);
```

**Description**

Test whether the stream data is memory-based.

**Parameter**

---

objStream	[In] The input stream object.
-----------	-------------------------------

---

**Return**

[TRUE](#) for the stream data being memory-based.

**Head file reference**

fpd\_objsTempl.h: 1278

**FPDStreamNew****Syntax**

```
FPD_Object FPDStreamNew (void );
```

**Description**

Creates a new empty stream object.

**Return**

A stream object.

**Head file reference**

fpd\_objsTempl.h: 1192

**Note:** Destroys the stream object through FPDOBJECTDESTROY();

**FPDStreamReadRawData**

**Syntax**

```
FS_BOOL FPDStreamReadRawData (
    FPD Object objStream,
    FS DWORD start_pos,
    FS LPBYTE pBuf,
    FS DWORD bufSize
);
```

**Description**

Reads raw data.

**Parameter**


---

objStream	[In] The input stream object.
-----------	-------------------------------

---

start_pos	[In] The start position in the stream data to read from.
-----------	--

---

pBuf	[Out] The buffer to receive the read data.
------	--

---

bufSize	[In] The size in bytes expected to read.
---------	--

---

**Return**

Non-zero means successful, otherwise failed.

**Head file reference**

fpd\_objsTempl.h: 1266

**FPDStreamSetData****Syntax**

```
void FPDStreamSetData (
    FPD Object objStream,
    FS LPBYTE pData,
    FS DWORD size,
    FS BOOL bCompressed
);
```

**Description**

Sets stream data. If *pData* is [NULL](#), just allocate stream buffer. The data can be uncompressed or compressed. If it's uncompressed, then previous filter info will be dropped (if any). If it's compressed, the caller should also maintain the filter information in the dictionary.

**Parameter**


---

objStream	[In] The input stream object.
-----------	-------------------------------

---

---

pData	[In] The stream data to set.
size	[In] The size in bytes of the stream data.
bCompressed	[In] Whether the data is compressed. param[in] bKeepBuf Whether the buffer will be maintained by the stream object.

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1218

## FPD\_StreamAcc

**[Return from](#) [Used by](#)**

### Description

Accessor of stream object: [FPD\\_StreamAcc](#) depends on a stream, it maintains a buffer for the data of a stream. This buffer may be temporary, if the stream is encoded by one or more data encoders. [FPD\\_StreamAcc](#) doesn't decode any image encoders. See [FPDStreamAccNew](#) , [FPDStreamAccDestroy](#) .

### Returned from

[FPDDocLoadFontFile](#)  
[FPDFontGetFontFile](#)  
[FPDStreamAccNew](#)

### Used by

[FPDStreamAccDestroy](#)  
[FPDStreamAccDetachData](#)  
[FPDStreamAccGetData](#)  
[FPDStreamAccGetDict](#)  
[FPDStreamAccGetImageDecoder](#)  
[FPDStreamAccGetImageParam](#)  
[FPDStreamAccGetSize](#)  
[FPDStreamAccGetStream](#)

## Functions

### Functions summary

**[FPDStreamAccDestroy](#)**

Destroys the accessor of stream object.

**[FPDStreamAccDetachData](#)**

Detaches the data buffer from this stream accessor. After this call, the caller is now responsible for releasing the data buffer.

#### [FPDStreamAccGetData](#)

Gets the loaded data pointer.

#### [FPDStreamAccGetDict](#)

Gets the dictionary of the stream object attached to.

#### [FPDStreamAccGetImageDecoder](#)

Gets the image decoder name.

#### [FPDStreamAccGetImageParam](#)

Gets the image params dictionary.

#### [FPDStreamAccGetSize](#)

Gets the loaded data size in bytes.

#### [FPDStreamAccGetStream](#)

Gets the stream object attached to.

#### [FPDStreamAccLoadAllData](#)

Must call this function to actually attach to a stream.

#### [FPDStreamAccNew](#)

Creates the accessor of stream object.

## Functions detail

### FPDStreamAccDestroy

#### **Syntax**

```
void FPDStreamAccDestroy (
    FPD StreamAcc stmAcc
);
```

#### **Description**

Destroys the accessor of stream object.

#### **Parameter**

---

stmAcc	[In] The input accessor of stream object.
--------	---

---

#### **Return**

void

#### **Head file reference**

fpd\_objsTempl.h: 1304

### FPDStreamAccDetachData

#### **Syntax**

```
FS_LPBYTE FPDStreamAccDetachData (
    FPD StreamAcc stmAcc
);
```

#### **Description**

Detaches the data buffer from this stream accessor. After this call, the caller is now responsible for releasing the data buffer.

**Parameter**

---

stmAcc	[In] The input accessor of stream object.
--------	---

---

**Return**

The data buffer from this stream accessor.

**Head file reference**

fpd\_objsTempl.h: 1362

**FPDStreamAccGetData****Syntax**

```
FS_LPCBYTE FPDStreamAccGetData (
    FPD_StreamAcc stmAcc
);
```

**Description**

Gets the loaded data pointer.

**Parameter**

---

stmAcc	[In] The input accessor of stream object.
--------	---

---

**Return**

The loaded data pointer.

**Head file reference**

fpd\_objsTempl.h: 1344

**FPDStreamAccGetDict****Syntax**

```
FPD_Object FPDStreamAccGetDict (
    FPD_StreamAcc stmAcc
);
```

**Description**

Gets the dictionary of the stream object attached to.

**Parameter**

---

stmAcc	[In] The input accessor of stream object.
--------	---

---

**Return**

The dictionary of the stream object attached to.

**Head file reference**

fpd\_objsTempl.h: 1335

**FPDStreamAccGetImageDecoder****Syntax**

```
char* FPDStreamAccGetImageDecoder (
    FPD_StreamAcc stmAcc
);
```

**Description**

Gets the image decoder name.

**Parameter**

---

stmAcc	[In] The input accessor of stream object.
--------	---

---

**Return**

The image decoder name.

**Head file reference**

fpd\_objsTempl.h: 1372

**FPDStreamAccGetImageParam****Syntax**

```
FPD_Object FPDStreamAccGetImageParam (
    FPD_StreamAcc stmAcc
);
```

**Description**

Gets the image params dictionary.

**Parameter**

---

stmAcc	[In] The input accessor of stream object.
--------	---

---

**Return**

The image params dictionary.

**Head file reference**

fpd\_objsTempl.h: 1381

## FPDStreamAccGetSize

### Syntax

```
FS_DWORD FPDStreamAccGetSize (
    FPD StreamAcc stmAcc
);
```

### Description

Gets the loaded data size in bytes.

### Parameter

---

stmAcc	[In] The input accessor of stream object.
--------	---

---

### Return

The loaded data size in bytes.

### Head file reference

fpd\_objsTempl.h: 1353

## FPDStreamAccGetStream

### Syntax

```
FPD_Object FPDStreamAccGetStream (
    FPD StreamAcc stmAcc
);
```

### Description

Gets the stream object attached to.

### Parameter

---

stmAcc	[In] The input accessor of stream object.
--------	---

---

### Return

The stream object attached to.

### Head file reference

fpd\_objsTempl.h: 1326

## FPDStreamAccLoadAllData

### Syntax

```
void FPDStreamAccLoadAllData (
    stmAcc,
```

---

```
FPD_Object obj,
FS_BOOL bRawAccess,
FS_DWORD estimated_size,
FS_BOOL bImageAcc
);
```

**Description**

Must call this function to actually attach to a stream.

**Parameter**

stmAcc	[In] The input accessor of stream object.
obj	[In] The stream object to be attached to.
bRawAccess	[In] Whether access the stream data rawly.
estimated_size	[In] The estimated size in bytes of the loaded stream data.
bImageAcc	[In] Whether access the image filter or not.

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 1313

**FPDStreamAccNew****Syntax**

```
FPD_StreamAcc FPDStreamAccNew (void );
```

**Description**

Creates the accessor of stream object.

**Return**

The accessor of stream object.

**Head file reference**

fpd\_objsTempl.h: 1295

**FPD\_StreamFilter**

## **Return from Used by**

### Description

Data filter created for accessing PDF stream data. See [FPDStreamFilterDestroy](#).

### Returned from

#### [FPDStreamGetStreamFilter](#)

### Used by

[FPDStreamFilterDestroy](#)  
[FPDStreamFilterGetSrcPos](#)  
[FPDStreamFilterGetStream](#)  
[FPDStreamFilterReadBlock](#)

## Functions

### Functions summary

#### [FPDStreamFilterDestroy](#)

Destroys the data filter created for accessing PDF stream data.

#### [FPDStreamFilterGetSrcPos](#)

Gets current source position (in the raw data stream).

#### [FPDStreamFilterGetStream](#)

Gets the stream object.

#### [FPDStreamFilterReadBlock](#)

Reads a data block. Return number of bytes actually read. If read size is less than the asked size, it indicates EOF.

### Functions detail

#### FPDStreamFilterDestroy

##### Syntax

```
void FPDStreamFilterDestroy (
    FPD\_StreamFilter flt
);
```

##### Description

Destroys the data filter created for accessing PDF stream data.

##### Parameter

---

flt	[In] The input data filter created for accessing PDF stream data.
-----	---

---

##### Return

void

**Head file reference**

fpd\_objsTempl.h: 1398

**FPDStreamFilterGetSrcPos****Syntax**

```
FS_DWORD FPDStreamFilterGetSrcPos (
    FPD\_StreamFilter flt
);
```

**Description**

Gets current source position (in the raw data stream).

**Parameter**

---

flt	[In] The input data filter created for accessing PDF stream data.
-----	---

---

**Return**

The current source position (in the raw data stream).

**Head file reference**

fpd\_objsTempl.h: 1419

**FPDStreamFilterGetStream****Syntax**

```
FPD_Object FPDStreamFilterGetStream (
    FPD\_StreamFilter flt
);
```

**Description**

Gets the stream object.

**Parameter**

---

flt	[In] The input data filter created for accessing PDF stream data.
-----	---

---

**Return**

The stream object.

**Head file reference**

fpd\_objsTempl.h: 1428

**FPDStreamFilterReadBlock****Syntax**

```
FS_DWORD FPDStreamFilterReadBlock (
    FPD\_StreamFilter flt,
    FS\_LPBYTE buffer,
    FS\_DWORD size
);
```

### Description

Reads a data block. Return number of bytes actually read. If read size is less than the asked size, it indicates EOF.

### Parameter

---

flt	[In] The input data filter created for accessing PDF stream data.
-----	---

---

buffer	[Out] It receives the read data.
--------	----------------------------------

---

size	[In] The size in bytes to read.
------	---------------------------------

---

### Return

The number of bytes actually read.

### Head file reference

fpd\_objsTempl.h: 1407

## FPD\_StreamWrite

### Description

stream writing interface.

## FPD\_String

### Description

A FPD\_String is a sequences of characters, enclosed in parentheses. *See Section 3.2.3 in the PDF Reference* for details.

### Functions

#### Functions summary

##### [FPDStringGetString](#)

Gets a ref to the data of the string object.

##### [FPDStringIdentical](#)

Compares with another string object.

##### [FPDStringIsHex](#)

Tests whether this string object has the hex flag.

**FPDStringNew**

Creates a string object from a byte string.

**FPDStringNewW**

Creates a string object from a wide string.

**FPDStringSetHex**

Changes the hex flag.

## Functions detail

### FPDStringGetString

**Syntax**

```
void FPDStringGetString (
    FPD Object objString,
    FS ByteString* outString
);
```

**Description**

Gets a ref to the data of the string object.

**Parameter**

---

objString	[In] The input string object.
-----------	-------------------------------

---

outString	[Out] It receives the ref to the data of the string object.
-----------	---

---

**Return**

void

**Head file reference**

fpd\_objsTempl.h: 440

### FPDStringIdentical

**Syntax**

```
FS_BOOL FPDStringIdentical (
    FPD Object objString,
    FPD Object other_objString
);
```

**Description**

Compares with another string object.

**Parameter**

---

objString	[In] The input string object.
-----------	-------------------------------

---

---

other_objString	[In] The other string object.
-----------------	-------------------------------

---

**Return**

Non-zero means identical, otherwise not identical.

**Head file reference**

fpd\_objsTempl.h: 450

**FPDStringIsHex****Syntax**

```
FS_BOOL FPDStringIsHex (
    FPD Object objString
);
```

**Description**

Tests whether this string object has the hex flag.

**Parameter**

---

objString	[In] The input string object.
-----------	-------------------------------

---

**Return**

[TRUE](#) for string object having the hex flag.

**Head file reference**

fpd\_objsTempl.h: 470

**FPDStringNew****Syntax**

```
FPD_Object FPDStringNew (
    FS ByteString str,
    FS\_BOOL bHex
);
```

**Description**

Creates a string object from a byte string.

**Parameter**

---

str	[In] The input byte string.
-----	-----------------------------

---

---

bHex	[In] The input hex flag.
------	--------------------------

---

**Return**

A string object.

**Head file reference**

fpd\_objsTempl.h: 419

**Related method**

[FPDOObjectDestroy](#)

**FPDStringNewW****Syntax**

```
FPD_Object FPDStringNewW (
    FS\_LPCWSTR wstr
);
```

**Description**

Creates a string object from a wide string.

**Parameter**

---

wstr	[In] The input wide string.
------	-----------------------------

---

**Return**

A string object.

**Head file reference**

fpd\_objsTempl.h: 430

**Related method**

[FPDOObjectDestroy](#)

**FPDStringSetHex****Syntax**

```
void FPDStringSetHex (
    FPD\_Object objString,
    FS\_BOOL bHex
);
```

**Description**

Changes the hex flag.

**Parameter**

---

objString	[In] The input string object.
-----------	-------------------------------

---

bHex	[In] The input hex flag.
------	--------------------------

---



**Return**

void

**Head file reference**

fpd\_objsTempl.h: 460

## FPD\_SubstFont

**Return from Used by****Description**

Substitution font. See [FPDSubstFontNew](#) , [FPDSubstFontDestroy](#) .

**Returned from****FPDFontGetSubstFont****Used by****FPDFontGetSubstFontCharset****FPDFontGetSubstFontWeight**

## FPD\_TextObject

**Return from****Description**

A FPD\_TextObject is a graphic object (a subclass of FPD\_PageObject) representing one or more character strings on a page. A text object consists of one or more character strings that identify sequences of glyphs to be painted. Like a path, text can be stroked, filled, or used as a clipping boundary.

**Returned from****FPDTextObjectGetItemInfo****Functions****Functions summary****FPDTextObjectCalcCharPos**

Calculates origin positions in text space, for each character.

**FPDTextObjectCountChars**

Gets the count of characters in the text object.

**FPDTextObjectCountItems**

Gets the count of text object items.

**FPDTextObjectDestroy**

Destroys the input PDF text object. If it is added to the page, it is taken over and don't destroy it.

**FPDTextObjectGetCharInfo**

Gets the information of specified character.

**FPDTextObjectGetData**

Gets text data.

**FPDTextObjectGetFont**

Gets the font.

**FPDTextObjectGetFontSize**

Gets the font size.

**FPDTextObjectGetItemInfo**

Gets specified text object item information.

**FPDTextObjectGetPosX**

Gets the x-coordinate of the origin in the device space

**FPDTextObjectGetPosY**

Gets the y-coordinate of the origin in the device space.

**FPDTextObjectGetTextMatrix**

Gets matrix from text space to object space.

**FPDTextObjectNew**

Creates a new empty PDF text object.

**FPDTextObjectRecalcPositionData**

Recalculates the position data.

**FPDTextObjectSetData**

Sets text data.

**FPDTextObjectSetEmpty**

Sets the text object to be empty.

**FPDTextObjectSetPosition**

Sets the origin position in device space.

**FPDTextObjectSetText**

Sets a single text segment without any kerning inside.

**FPDTextObjectSetText2**

Sets text using segmented fashion.

**FPDTextObjectSetText3**

Sets text using char-kerning pair fashion.

**FPDTextObjectSetTextState**

Sets the text state.

**FPDTextObjectTransform**

Transforms the text object.

## Functions detail

### FPDTextObjectCalcCharPos

#### Syntax

```
void FPDTextObjectCalcCharPos (
    FPD\_PageObject objText,
    FS\_FLOAT* outPosArray
);
```

#### Description

Calculates origin positions in text space, for each character.

**Parameter**

objText	[In] The input PDF text object.
outPosArray	[Out] It receives the character position array.

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1437

**Note:** The position array must be allocated and freed by caller. It must contain at least nChars\*2 elements. For each character, the origin position (along the text baseline) and next origin position will be calculated.

**FPDTextObjectCountChars****Syntax**

```
FS_INT32 FPDTextObjectCountChars (
    FPD\_PageObject objText
);
```

**Description**

Gets the count of characters in the text object.

**Parameter**

objText	[In] The input PDF text object.
---------	---------------------------------

**Return**

The count of characters in the text object.

**Head file reference**

fpd\_pageobjTempl.h: 1296

**FPDTextObjectCountItems****Syntax**

```
FS_INT32 FPDTextObjectCountItems (
    FPD\_PageObject objText
);
```

**Description**

Gets the count of text object items.

#### Parameter

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

#### Return

The count of text object items.

#### Head file reference

fpd\_pageobjTempl.h: 1277

### FPDTextObjectDestroy

#### Syntax

```
void FPDTextObjectDestroy (
    FPD\_PageObject objText
);
```

#### Description

Destroys the input PDF text object. If it is added to the page, it is taken over and don't destroy it.

#### Parameter

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

#### Return

void

#### Head file reference

fpd\_pageobjTempl.h: 1268

### FPDTextObjectGetCharInfo

#### Syntax

```
void FPDTextObjectGetCharInfo (
    FPD\_PageObject objText,
    FS\_INT32 index,
    FS\_DWORD* outCharcode,
    FS\_FLOAT* outKerning
);
```

#### Description

Gets the information of specified character.

#### Parameter



---

objText	[In] The input PDF text object.
index	[In] Specifies zero-based character index in the text object.
outCharcode	[Out] It receives the character code.
outKerning	[Out] It receives the kerning(x-direction only).

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1305

**FPDTextObjectGetData****Syntax**

```
void FPDTextObjectGetData (
    FPD\_PageObject objText,
    FS\_INT32* outCharCount,
    FS\_DWORD** outCharCodes,
    FS\_FLOAT** outCharPos
);
```

**Description**

Gets text data.

**Parameter**


---

objText	[In] The input PDF text object.
outCharCount	[Out] It receives the count of characters.
outCharCodes	[Out] It receives the character codes array.
outCharPos	[Out] It receives the character positions array in text space.

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1462

## FPDTextObjectGetFont

### Syntax

```
FPD_Font FPDTextObjectGetFont (
    FPD\_PageObject objText
);
```

### Description

Gets the font.

### Parameter

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

### Return

The font.

### Head file reference

`fpd_pageobjTempl.h: 1345`

## FPDTextObjectGetFontSize

### Syntax

```
FS_FLOAT FPDTextObjectGetFontSize (
    FPD\_PageObject objText
);
```

### Description

Gets the font size.

### Parameter

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

### Return

The font size.

### Head file reference

`fpd_pageobjTempl.h: 1354`

## FPDTextObjectGetItemInfo

### Syntax

```
FPD_TextObjectItem FPDTextObjectGetItemInfo (
    FPD\_PageObject objText,
    FS_INT32 index
);
```



**Description**

Gets specified text object item information.

**Parameter**

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

index	[In] Specifies zero-based item index in the text object.
-------	--

---

**Return**

The specified text object item information.

**Head file reference**

fpd\_pageobjTempl.h: 1286

**FPDTextObjectGetPosX****Syntax**

```
FS_FLOAT FPDTextObjectGetPosX (
    FPD\_PageObject objText
);
```

**Description**

Gets the x-coordinate of the origin in the device space

**Parameter**

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

**Return**

The x-coordinate of the origin in the device space

**Head file reference**

fpd\_pageobjTempl.h: 1317

**FPDTextObjectGetPosY****Syntax**

```
FS_FLOAT FPDTextObjectGetPosY (
    FPD\_PageObject objText
);
```

**Description**

Gets the y-coordinate of the origin in the device space.

**Parameter**

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

**Return**

The y-coordinate of the origin in the device space.

**Head file reference**

fpd\_pageobjTempl.h: 1326

**FPDTextObjectGetTextMatrix****Syntax**

```
void FPDTextObjectGetTextMatrix (
    FPD\_PageObject objText,
    FS\_AffineMatrix* outMatrix
);
```

**Description**

Gets matrix from text space to object space.

**Parameter**

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

---

outMatrix	[Out] It receives the matrix from text space to object space.
-----------	---

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1335

**FPDTextObjectNew****Syntax**

```
FPD_PageObject FPDTextObjectNew (void );
```

**Description**

Creates a new empty PDF text object.

**Return**

The PDF text object

**Head file reference**

fpd\_pageobjTempl.h: 1259

### FPDTextObjectRecalcPositionData

#### Syntax

```
void FPDTextObjectRecalcPositionData (
    FPD\_PageObject objText
);
```

#### Description

Recalculates the position data.

#### Parameter

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

#### Return

void

#### Head file reference

fpd\_pageobjTempl.h: 1474

### FPDTextObjectSetData

#### Syntax

```
void FPDTextObjectSetData (
    FPD\_PageObject objText,
    FS\_INT32 nChars,
    FS\_DWORD* pCharCodes,
    FS\_FLOAT* pCharPos,
    FS\_FLOAT x,
    FS\_FLOAT y
);
```

#### Description

Sets text data.

#### Parameter

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

nChars	[In] The count of characters to set.
--------	--------------------------------------

---

pCharCodes	[In] The input character codes array.
------------	---------------------------------------

---



---

pCharPos	[In] The input character positions array in text space.
----------	---

---

x	[In] The x-coordinate of the origin position, in device space.
---	--

---

y	[In] The y-coordinate of the origin position, in device space.
---	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1448

**FPDTextObjectSetEmpty****Syntax**

```
void FPDTextObjectSetEmpty (
    FPD\_PageObject objText
);
```

**Description**

Sets the text object to be empty.

**Parameter**

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1363

**FPDTextObjectSetPosition****Syntax**

```
void FPDTextObjectSetPosition (
    FPD\_PageObject objText,
    FS\_FLOAT x,
    FS\_FLOAT y
);
```

**Description**

Sets the origin position in device space.

**Parameter**

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

x	[In] The x-coordinate in device space.
---	--

---

y	[In] The y-coordinate in device space.
---	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1406

**FPDTextObjectSetText****Syntax**

```
void FPDTextObjectSetText (
    FPD\_PageObject objText,
    const FS\_ByteString strText
);
```

**Description**

Sets a single text segment without any kerning inside.

**Parameter**


---

objText	[In] The input PDF text object.
---------	---------------------------------

---

strText	[In] The input text segment.
---------	------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1372

**FPDTextObjectSetText2****Syntax**

```
void FPDTextObjectSetText2 (
    FPD\_PageObject objText,
    FS\_ByteStringArray strTextArr,
    FS\_FLOAT* pKerning
);
```

**Description**

Sets text using segmented fashion.

**Parameter**

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

---

strTextArr	[In] The input text segments.
------------	-------------------------------

---

---

pKerning	[In] The kerning array.
----------	-------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1382

**Since**

[SDK LATEEST VERSION > 2.1.0.4](#)

**FPDTextObjectSetText3****Syntax**

```
void FPDTextObjectSetText3 (
    FPD\_PageObject objText,
    FS\_INT32 nChars,
    FS\_DWORD* pCharCodes,
    FS\_FLOAT* pKernings
);
```

**Description**

Sets text using char-kerning pair fashion.

**Parameter**

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

---

nChars	[In] The count of input character codes.
--------	--

---

---

pCharCodes	[In] The input character codes.
------------	---------------------------------

---

---

pKernings	[In] The input kerning array.
-----------	-------------------------------

---

**Return**

void



**Head file reference**

fpd\_pageobjTempl.h: 1394

**FPDTextObjectSetTextState****Syntax**

```
void FPDTextObjectSetTextState (
    FPD\_PageObject objText,
    FPD\_TextState textState
);
```

**Description**

Sets the text state.

**Parameter**

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

textState	[In] The new text state.
-----------	--------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1417

**FPDTextObjectTransform****Syntax**

```
void FPDTextObjectTransform (
    FPD\_PageObject objText,
    FS\_AffineMatrix matrix
);
```

**Description**

Transforms the text object.

**Parameter**

---

objText	[In] The input PDF text object.
---------	---------------------------------

---

matrix	[In] The transformation matrix.
--------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 1427

## FPD\_TextPage

**[Return from Used by](#)**

### Description

Text page for PDF text processing. See [FPDTextNew](#) , [FPDTextDestroy](#) .

### Returned from

[FPDTextPageFindNew](#)  
[FPDTextPageNew](#)

### Used by

[FPDTextPageFindDestroy](#)  
[FPDTextPageFindFindFirst](#)  
[FPDTextPageFindFindNext](#)  
[FPDTextPageFindFindPrev](#)  
[FPDTextPageFindGetCurOrder](#)  
[FPDTextPageFindGetMatchedCount](#)  
[FPDTextPageFindGetRectArray](#)  
[FPDTextPageCountBoundedSegments](#)  
[FPDTextPageCountChars](#)  
[FPDTextPageCountRects](#)  
[FPDTextPageDestroy](#)  
[FPDTextPageGetBoundedSegment](#)  
[FPDTextPageGetCharInfo](#)  
[FPDTextPageGetIndexAtPos](#)  
[FPDTextPageGetOrderByDirection](#)  
[FPDTextPageGetPageText](#)  
[FPDTextPageGetRect](#)  
[FPDTextPageGetRectArray](#)  
[FPDTextPageGetRectsArrayByRect](#)  
[FPDTextPageGetTextByRect](#)  
[FPDTextPageGetWordBreak](#)  
[FPDTextPageIsParsed](#)  
[FPDTextPageParseTextPage](#)

### Definitions

#### Definitions summary

**[FPD CHAR ERROR](#)**

error.

**[FPD CHAR GENERATED](#)**

generated.

**FPD\_CHAR\_NORMAL**

normal.

**FPD\_CHAR\_UNICODE**

ununicode.

**FPD\_TEXT\_DOWN**

Down direction.

**FPD\_TEXT\_LEFT**

Left direction.

**FPD\_TEXT\_RIGHT**

Right direction.

**FPD\_TEXT\_UP**

Up direction.

**FPD\_TEXT\_DISPLAY\_ORDER**

Display(appearance) order.

**FPD\_TEXT\_STREAM\_ORDER**

Stream order.

## Definitions detail

### FPD\_CHAR\_ERROR

**Syntax**

```
#define FPD_CHAR_ERROR -1
```

**Description**

error.

**Group**

[FPDCharInfoFlag](#)

**Head file reference**

fpd\_textExpT.h: 114

### FPD\_CHAR\_GENERATED

**Syntax**

```
#define FPD_CHAR_GENERATED 1
```

**Description**

generated.

**Group**

[FPDCharInfoFlag](#)

**Head file reference**

fpd\_textExpT.h: 118

## FPD\_CHAR\_NORMAL

**Syntax**

```
#define FPD_CHAR_NORMAL 0
```

**Description**

normal.

**Group**

[FPDCharInfoFlag](#)

**Head file reference**

fpd\_textExpT.h: 116

## FPD\_CHAR\_UNUNICODE

**Syntax**

```
#define FPD_CHAR_UNUNICODE 2
```

**Description**

ununicode.

**Group**

[FPDCharInfoFlag](#)

**Head file reference**

fpd\_textExpT.h: 120

## FPD\_TEXT\_DOWN

**Syntax**

```
#define FPD_TEXT_DOWN 2
```

**Description**

Down direction.

**Group**

[FPDTextDirectionsType](#)

**Head file reference**

fpd\_textExpT.h: 183

## FPD\_TEXT\_LEFT

**Syntax**

```
#define FPD_TEXT_LEFT -1
```

**Description**

Left direction.

**Group**

[FPDTextDirectionsType](#)

**Head file reference**

fpd\_textExpT.h: 177

## FPD\_TEXT\_RIGHT

**Syntax**

#define FPD\_TEXT\_RIGHT 1

**Description**

Right direction.

**Group**

[FPDTextDirectionsType](#)

**Head file reference**

fpd\_textExpT.h: 179

## FPD\_TEXT\_UP

**Syntax**

#define FPD\_TEXT\_UP -2

**Description**

Up direction.

**Group**

[FPDTextDirectionsType](#)

**Head file reference**

fpd\_textExpT.h: 181

## FPD\_TEXT\_DISPLAY\_ORDER

**Syntax**

#define FPD\_TEXT\_DISPLAY\_ORDER 1

**Description**

Display(appearance) order.

**Group**

[FPDTextPageFlags](#)**Head file reference**

fpd\_textExpT.h: 165

**FPD\_TEXT\_STREAM\_ORDER****Syntax**

#define FPD\_TEXT\_STREAM\_ORDER 0

**Description**

Stream order.

**Group**[FPDTextPageFlags](#)**Head file reference**

fpd\_textExpT.h: 162

## Structures

### Structures summary

[FPD\\_CHAR\\_INFO](#)

Character info for TextPage.

### Structs detail

[FPD\\_CHAR\\_INFO](#)**Syntax**

```
typedef struct {
    FS_WCHAR m_Uncode,
    FS_BOOL m_Flag,
    FS_FLOAT m_FontSize,
    FS_FLOAT m-OriginX,
    FS_FLOAT m-OriginY,
    FS_FloatRect m_CharBox,
    FPD_PageObject m_pTextObj,
    FS_AffineMatrix m_Matrix
} FPD_CHAR_INFO;
```

**Description**

Character info for TextPage.

**Head file reference**

fpd\_textExpT.h: 127

**m\_Uncode**

Unicode for the character.

**m\_Flag**

Character's flag.

**m\_FontSize**

The font size.

**m-OriginX**

The x-coordinate of the origin position.

**m-OriginY**

The y-coordinate of the origin position.

**m\_CharBox**

The character box. int page space.

**m\_pTextObj**

Which text object it belongs to.

**m\_Matrix**

The matrix.

## Functions

### Functions summary

**FPDTextPageCountBoundedSegments**

Gets the count of segment in specified rectangle.

**FPDTextPageCountChars**

Gets the count of character in the text page.

**FPDTextPageCountRects**

Gets the count of text rectangle in the text page.

**FPDTextPageDestroy**

Destroys the text page.

**FPDTextPageGetBoundedSegment**

Gets the specified bounded segment.

**FPDTextPageGetCharInfo**

Gets the specified character information.

**FPDTextPageGetIndexAtPos**

If there are no character in this point, the return value *xTolerance* or *yTolerance* is between 0 and 30.

**FPDTextPageGetOrderByDirection**

Gets the character index by specific direction. if the return value is -1 means previous page,-2 for next page.

**FPDTextPageGetPageText**

Gets the text of a page.

**FPDTextPageGetRect**

Gets the specified text rectangle in the text page.

**FPDTextPageGetRectArray**

Gets the rectangle array for character index interzone. [start, start+count)

**FPDTextPageGetRectsArrayByRect**

Gets the text rectangle array by a rectangle.

**FPDTextPageGetTextByRect**

Gets the text in specified rectangle.

**FPDTextPageGetWordBreak**

Not support now.

**FPDTextPageIsParsed**

Check whether the text page has been parsed.

**FPDTextPageNew**

Creates a new text page for PDF text processing.

**FPDTextPageParseTextPage**

Parses the pdf page. The page object may have parsed content objects.

**Functions detail****FPDTextPageCountBoundedSegments****Syntax**

```
FS_INT32 FPDTextPageCountBoundedSegments (
    FPD_TextPage page,
    FS_FLOAT left,
    FS_FLOAT top,
    FS_FLOAT right,
    FS_FLOAT bottom
);
```

**Description**

Gets the count of segment in specified rectangle.

**Parameter**

page	[In] The input text page.
left	[In] Left position of the text coordinate.
top	[In] Top position of the text coordinate.
right	[In] Right position of the text coordinate.
bottom	[In] Bottom position of the text coordinate.

**Return**

The count of segment in specified rectangle.

**Head file reference**

fpd\_textTempl.h: 265

## FPDTextPageCountChars

### Syntax

```
FS_INT32 FPDTextPageCountChars (
    FPD\_TextPage page
);
```

### Description

Gets the count of character in the text page.

### Parameter

---

page	[In] The input text page.
------	---------------------------

---

### Return

The count of character in the text page.

### Head file reference

[fpd\\_textTempl.h](#): 165

## FPDTextPageCountRects

### Syntax

```
FS_INT32 FPDTextPageCountRects (
    FPD\_TextPage page,
    FS_INT32 start,
    FS_INT32 nCount
);
```

### Description

Gets the count of text rectangle in the text page.

### Parameter

---

page	[In] The input text page.
------	---------------------------

---

---

start	[In] The index of the starting character.
-------	---

---

---

nCount	[In] The character's count.
--------	-----------------------------

---

### Return

The count of text rectangle.

### Head file reference

[fpd\\_textTempl.h](#): 244

## FPDTextPageDestroy

### Syntax

```
void FPDTextPageDestroy (
    FPD\_TextPage page
);
```

### Description

Destroys the text page.

### Parameter

---

page	[In] The input text page.
------	---------------------------

### Return

void

### Head file reference

[fpd\\_textTempl.h](#): 138

## FPDTextPageGetBoundedSegment

### Syntax

```
void FPDTextPageGetBoundedSegment (
    FPD\_TextPage page,
    FS\_INT32 index,
    FS\_INT32* outstart,
    FS\_INT32* outcount
);
```

### Description

Gets the specified bounded segment.

### Parameter

---

page	[In] The input text page.
------	---------------------------

index	[In] The specified index of bounded segment.
-------	--

outstart	[In/Out] The index of starting character.
----------	---

outcount	[In/Out] The count of characters.
----------	-----------------------------------

### Return

void

**Head file reference**

fpd\_textTempl.h: 278

**FPDTextPageGetCharInfo****Syntax**

```
void FPDTextPageGetCharInfo (
    FPD\_TextPage page,
    FS\_INT32 index,
    FPD\_CHAR\_INFO* outInfo
);
```

**Description**

Gets the specified character information.

**Parameter**

page	[In] The input text page.
index	[In] The index of the character.
outInfo	[Out] It receives the information of the character.

**Return**

void

**Head file reference**

fpd\_textTempl.h: 174

**FPDTextPageGetIndexAtPos****Syntax**

```
FS\_INT32 FPDTextPageGetIndexAtPos (
    FPD\_TextPage page,
    FS\_FLOAT x,
    FS\_FLOAT y,
    FS\_FLOAT xTolerance,
    FS\_FLOAT yTolerance
);
```

**Description**

If there are no character in this point, the return value *xTolerance* or *yTolerance* is between 0 and 30.

**Parameter**

---

page	[In] The input text page.
------	---------------------------

---

x	[In] The x coordinate of the point.
---	-------------------------------------

---

y	[In] The y coordinate of the point.
---	-------------------------------------

---

xTolerance	[In] X direction recommended value.
------------	-------------------------------------

---

yTolerance	[In] Y direction recommended value.
------------	-------------------------------------

---

**Return**

The index of specified char.

**Head file reference**

fpd\_textTempl.h: 197

**FPDTextPageGetOrderByDirection****Syntax**

```
FS_INT32 FPDTextPageGetOrderByDirection (
    FPD\_TextPage page,
    FS_INT32 order,
    FS_INT32 direction
);
```

**Description**

Gets the character index by specific direction. if the return value is -1 means previous page,-2 for next page.

**Parameter**


---

page	[In] The input text page.
------	---------------------------

---

order	[In] The character index.
-------	---------------------------

---

direction	[In] See directions type.
-----------	---------------------------

---

**Return**

The character with specified relationship of current character

**Head file reference**

fpd\_textTempl.h: 211

**FPDTextPageGetPageText****Syntax**

```
void FPDTextPageGetPageText (
    FPD_TextPage page,
    FS_INT32 start,
    FS_INT32 nCount,
    FS_WideString* outText
);
```

**Description**

Gets the text of a page.

**Parameter**


---

page	[In] The input text page.
start	[In] The start of the page text.
nCount	[In] The count of page text to get, -1 for the end of the page.
outText	[Out] It receives the text got.

---

**Return**

void

**Head file reference**

fpd\_textTempl.h: 301

**FPDTextPageGetRect****Syntax**

```
FS_FloatRect FPDTextPageGetRect (
    FPD_TextPage page,
    FS_INT32 rectIndex
);
```

**Description**

Gets the specified text rectangle in the text page.

**Parameter**


---

page	[In] The input text page.
rectIndex	[In] The specified rectangle index.

---

**Return**

The specified text rectangle in the text page.

**Head file reference**

fpd\_textTempl.h: 185

**FPDTextPageGetRectArray****Syntax**

```
void FPDTextPageGetRectArray (
    FPD\_TextPage page,
    FS\_INT32 start,
    FS\_INT32 nCount,
    FS\_FloatRectArray* outRectArray
);
```

**Description**

Gets the rectangle array for character index interzone. [start, start+count)

**Parameter**

---

page	[In] The input text page.
start	[In] The starting char index.
nCount	[In] Number of chars ( end char index - start char index).
outRectArray	[Out] It receives the rectangle array for character index interzone. [start, start+count).

---

**Return**

void

**Head file reference**

fpd\_textTempl.h: 185

**FPDTextPageGetRectsArrayByRect****Syntax**

```
void FPDTextPageGetRectsArrayByRect (
    FPD\_TextPage page,
    FS\_FloatRect rect,
    FS\_FloatRectArray* outRectArray
);
```

**Description**

Gets the text rectangle array by a rectangle.

**Parameter**

page	[In] The input text page.
rect	[In] The specified rectangle.
outRectArray	[Out] It receives the text rectangle array by a rectangle.

**Return**

void

**Head file reference**

fpd\_textTempl.h: 233

**FPDTextPageGetTextByRect****Syntax**

```
void FPDTextPageGetTextByRect (
    FPD\_TextPage page,
    FS\_FloatRect rect,
    FS\_WideString* outText
);
```

**Description**

Gets the text in specified rectangle.

**Parameter**

page	[In] The input text page.
rect	[In] The specified rectangle.
outText	[Out] It receives the text in specified rectangle.

**Return**

void

**Head file reference**

fpd\_textTempl.h: 222

**FPDTextPageGetWordBreak**

**Syntax**

```
FS_INT32 FPDTextPageGetWordBreak (
    FPD\_TextPage page,
    FS\_INT32 index,
    FS\_INT32 direction
);
```

**Description**

Not support now.

**Parameter**

---

page	[In] The input text page.
------	---------------------------

---

index	[In] The specified index.
-------	---------------------------

---

direction	[In] The input direction.
-----------	---------------------------

---

**Return**

The word break.

**Head file reference**

fpd\_textTempl.h: 290

**FPDTextPageIsParsed****Syntax**

```
FS_BOOL FPDTextPageIsParsed (
    FPD\_TextPage page
);
```

**Description**

Check whether the text page has been parsed.

**Parameter**

---

page	[In] The input text page.
------	---------------------------

---

**Return**

[TRUE](#) for the text page having been parsed, otherwise [FALSE](#).

**Head file reference**

fpd\_textTempl.h: 156

**FPDTextPageNew**

**Syntax**

```
FPD_TextPage FPDTextPageNew (
    FPD\_Page page,
    FS\_INT32 flags
);
```

**Description**

Creates a new text page for PDF text processing.

**Parameter**

page	[In] The input PDF page.
flags	[In] See <a href="#">FPDTextPageFlags</a> .

**Return**

A new text page for PDF text processing.

**Head file reference**

fpd\_textTempl.h: 128

**FPDTextPageParseTextPage****Syntax**

```
FS_BOOL FPDTextPageParseTextPage (
    FPD\_TextPage page
);
```

**Description**

Parses the pdf page. The page object may have parsed content objects.

**Parameter**

page	[In] The input text page.
------	---------------------------

**Return**

[TRUE](#) means success, otherwise [FALSE](#) .

**Head file reference**

fpd\_textTempl.h: 147

**FPD\_TextPageFind****[Return from Used by](#)**

## Description

To find a text in a page. See [FPDTextPageFindNew](#) , [FPDTextPageFindDestroy](#) .

### Returned from

[FPDTextPageFindNew](#)

### Used by

[FPDTextPageFindDestroy](#)  
[FPDTextPageFindFindFirst](#)  
[FPDTextPageFindFindNext](#)  
[FPDTextPageFindFindPrev](#)  
[FPDTextPageFindGetCurOrder](#)  
[FPDTextPageFindGetMatchedCount](#)  
[FPDTextPageFindGetRectArray](#)

## Functions

### Functions summary

[FPDTextPageFindDestroy](#)

Destroys the text searching object.

[FPDTextPageFindFindFirst](#)

Finds the first match with specified search flags, optionally with a start position.

[FPDTextPageFindFindNext](#)

Finds the next match.

[FPDTextPageFindFindPrev](#)

Finds the previous match.

[FPDTextPageFindGetCurOrder](#)

Gets the current order.

[FPDTextPageFindGetMatchedCount](#)

Gets the count of current matches.

[FPDTextPageFindGetRectArray](#)

Gets the matched rectangle array.

[FPDTextPageFindNew](#)

Creates text searching object from a text page.

### Functions detail

[FPDTextPageFindDestroy](#)

#### Syntax

```
void FPDTextPageFindDestroy (
    FPD\_TextPageFind find
);
```

#### Description

Destroys the text searching object.

**Parameter**


---

find	[In] The input text searching object.
------	---------------------------------------

---

**Return**

void

**Head file reference**

fpd\_textTempl.h: 330

**FPDTextPageFindFindFirst****Syntax**

```
FS_BOOL FPDTextPageFindFindFirst (
    FPD\_TextPageFind find,
    FS\_LPWSTR findwhat,
    FS\_INT32 flags,
    FS\_INT32 startPos
);
```

**Description**

Finds the first match with specified search flags, optionally with a start position.

**Parameter**


---

find	[In] The input text searching object.
------	---------------------------------------

---

findwhat	[In] What you want to find.
----------	-----------------------------

---

flags	[In] The search flags.
-------	------------------------

---

startPos	[In] The start position.
----------	--------------------------

---

**Return**[TRUE](#) means the first match is found, otherwise [FALSE](#).**Head file reference**

fpd\_textTempl.h: 339

**FPDTextPageFindFindNext****Syntax**

```
FS_BOOL FPDTextPageFindFindNext (
    FPD\_TextPageFind find
);
```

**Description**

Finds the next match.

**Parameter**

---

find	[In] The input text searching object.
------	---------------------------------------

---

**Return**

TRUE means the next match is found, otherwise not.

**Head file reference**

fpd\_textTempl.h: 351

**FPDTextPageFindFindPrev****Syntax**

```
FS_BOOL FPDTextPageFindFindPrev (
    FPD\_TextPageFind find
);
```

**Description**

Finds the previous match.

**Parameter**

---

find	[In] The input text searching object.
------	---------------------------------------

---

**Return**

TRUE means the previous match is found, otherwise not.

**Head file reference**

fpd\_textTempl.h: 360

**FPDTextPageFindGetCurOrder****Syntax**

```
FS_INT32 FPDTextPageFindGetCurOrder (
    FPD\_TextPageFind find
);
```

**Description**

Gets the current order.

**Parameter**

---

find	[In] The input text searching object.
------	---------------------------------------

---

**Return**

The current order.

**Head file reference**

fpd\_textTempl.h: 379

**FPDTextPageFindGetMatchedCount****Syntax**

```
FS_INT32 FPDTextPageFindGetMatchedCount (
    FPD\_TextPageFind find
);
```

**Description**

Gets the count of current matches.

**Parameter**

---

find	[In] The input text searching object.
------	---------------------------------------

---

**Return**

The count of current matches.

**Head file reference**

fpd\_textTempl.h: 388

**FPDTextPageFindGetRectArray****Syntax**

```
void FPDTextPageFindGetRectArray (
    FPD\_TextPageFind find,
    FS\_FloatRectArray* outRects
);
```

**Description**

Gets the matched rectangle array.

**Parameter**

---

find	[In] The input text searching object.
------	---------------------------------------

---

---

outRects	[Out] It receives the matched rectangle array.
----------	--

---

**Return**

void

**Head file reference**

fpd\_textTempl.h: 369

**FPDTextPageFindNew****Syntax**

```
FPD_TextPageFind FPDTextPageFindNew (
    const FPD_TextPage textpage
);
```

**Description**

Creates text searching object from a text page.

**Parameter**

---

textpage	[In] The input text page object.
----------	----------------------------------

---

**Return**

The text searching object.

**Head file reference**

fpd\_textTempl.h: 321

## FPD\_TextState

**Return from Used by****Description**

The text state for page rendering. See [FPDTextStateNew](#) , [FPDTextStateDestroy](#) .

**Returned from**

[FPDPageObjectGetTextState](#)  
[FPDTextStateNew](#)

**Used by**

[FPDPageArchiveLoaderLoadTextState](#)  
[FPDPageArchiveSaverSaveTextState](#)  
[FPDPageObjectSetTextState](#)  
[FPDTextObjectSetTextState](#)  
[FPDTextStateDestroy](#)  
[FPDTextStateGetBaselineAngle](#)  
[FPDTextStateGetCharSpace](#)  
[FPDTextStateGetFont](#)

[\*\*FPDTextStateGetFontSize\*\*](#)  
[\*\*FPDTextStateGetFontSizeH\*\*](#)  
[\*\*FPDTextStateGetFontSizeV\*\*](#)  
[\*\*FPDTextStateGetMatrix\*\*](#)  
[\*\*FPDTextStateGetModify\*\*](#)  
[\*\*FPDTextStateGetShearAngle\*\*](#)  
[\*\*FPDTextStateGetTextCTM\*\*](#)  
[\*\*FPDTextStateGetTextMode\*\*](#)  
[\*\*FPDTextStateGetWordSpace\*\*](#)  
[\*\*FPDTextStateIsNull\*\*](#)  
[\*\*FPDTextStateSetCharSpace\*\*](#)  
[\*\*FPDTextStateSetFont\*\*](#)  
[\*\*FPDTextStateSetFontSize\*\*](#)  
[\*\*FPDTextStateSetMatrix\*\*](#)  
[\*\*FPDTextStateSetTextCTM\*\*](#)  
[\*\*FPDTextStateSetTextMode\*\*](#)  
[\*\*FPDTextStateSetWordSpace\*\*](#)

## Functions

### Functions summary

#### [\*\*FPDTextStateDestroy\*\*](#)

Destroys the PDF text state object.

#### [\*\*FPDTextStateGetBaselineAngle\*\*](#)

Gets the angle between device space X-axis and text baseline. In radians.

#### [\*\*FPDTextStateGetCharSpace\*\*](#)

Gets the character space.

#### [\*\*FPDTextStateGetFont\*\*](#)

Gets the font.

#### [\*\*FPDTextStateGetFontSize\*\*](#)

Gets the font size.

#### [\*\*FPDTextStateGetFontSizeH\*\*](#)

Gets the horizontal size in device units

#### [\*\*FPDTextStateGetFontSizeV\*\*](#)

Gets the vertical size in device units.

#### [\*\*FPDTextStateGetMatrix\*\*](#)

Gets The text transformation matrix. Tt has 4 items.(matrix[4]).

#### [\*\*FPDTextStateGetModify\*\*](#)

The interface helps init the object if the object is NULL.

#### [\*\*FPDTextStateGetShearAngle\*\*](#)

Gets the angle that text space Y-axis shears in device space. In radians.

#### [\*\*FPDTextStateGetTextCTM\*\*](#)

Gets the CTM for stroking purpose. The CTM array must be allocated and freed by caller.  
It must contain 4 elements.

#### [\*\*FPDTextStateGetTextMode\*\*](#)

Gets the text mode.

#### [\*\*FPDTextStateGetWordSpace\*\*](#)

Gets the word space.

#### [\*\*FPDTextStateIsNull\*\*](#)

Tests whether the text state object is [\*\*NULL\*\*](#) or not.

#### [\*\*FPDTextStateNew\*\*](#)

Creates a new empty PDF text state object.

**FPDTextStateSetCharSpace**

Sets the character space.

**FPDTextStateSetFont**

Sets the font.

**FPDTextStateSetFontSize**

Sets the font size.

**FPDTextStateSetMatrix**

Sets the text transformation matrix.

**FPDTextStateSetTextCTM**

Sets the CTM for stroking purpose.

**FPDTextStateSetTextMode**

Sets the text mode.

**FPDTextStateSetWordSpace**

Sets the word space.

## Functions detail

### FPDTextStateDestroy

**Syntax**

```
void FPDTextStateDestroy (
    FPD_TextState textState
);
```

**Description**

Destroys the PDF text state object.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 489

### FPDTextStateGetBaselineAngle

**Syntax**

```
FS_FLOAT FPDTextStateGetBaselineAngle (
    FPD_TextState textState
);
```

**Description**

Gets the angle between device space X-axis and text baseline. In radians.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

**Return**

The angle between device space X-axis and text baseline. In radians.

**Head file reference**

fpd\_pageobjTempl.h: 553

**FPDTextStateGetCharSpace****Syntax**

```
FS_FLOAT FPDTextStateGetCharSpace (
    FPD\_TextState textState
);
```

**Description**

Gets the character space.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

**Return**

The character space.

**Head file reference**

fpd\_pageobjTempl.h: 651

**FPDTextStateGetFont****Syntax**

```
FPD_Font FPDTextStateGetFont (
    FPD\_TextState textState
);
```

**Description**

Gets the font.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

**Return**

The font.

**Head file reference**

fpd\_pageobjTempl.h: 498

**FPDTextStateGetFontSize****Syntax**

```
FS_FLOAT FPDTextStateGetFontSize (
    FPD\_TextState textState
);
```

**Description**

Gets the font size.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

**Return**

The font size.

**Head file reference**

fpd\_pageobjTempl.h: 517

**FPDTextStateGetFontSizeH****Syntax**

```
FS_FLOAT FPDTextStateGetFontSizeH (
    FPD\_TextState textState
);
```

**Description**

Gets the horizontal size in device units

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

**Return**

The horizontal size in device units

**Head file reference**

fpd\_pageobjTempl.h: 544

**FPDTextStateGetFontSizeV**

**Syntax**

```
FS_FLOAT FPDTextStateGetFontSizeV (
    FPD\_TextState textState
);
```

**Description**

Gets the vertical size in device units.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

**Return**

The vertical size in device units.

**Head file reference**

fpd\_pageobjTempl.h: 535

**FPDTextStateGetMatrix****Syntax**

```
FS_FLOAT* FPDTextStateGetMatrix (
    FPD\_TextState textState
);
```

**Description**

Gets The text transformation matrix. Tt has 4 items.(matrix[4]).

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

**Return**

The text transformation matrix. It has 4 items.(matrix[4]).

**Head file reference**

fpd\_pageobjTempl.h: 526

**FPDTextStateGetModify****Syntax**

```
void FPDTextStateGetModify (
    FPD\_TextState textState
);
```

**Description**

The interface helps init the object if the object is NULL.

#### Parameter

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

#### Return

void.

#### Head file reference

fpd\_pageobjTempl.h: 678

#### Since

[SDK LATEEST VERSION > 1.0](#)

## FPDTextStateGetShearAngle

#### Syntax

```
FS_FLOAT FPDTextStateGetShearAngle (
    FPD\_TextState textState
);
```

#### Description

Gets the angle that text space Y-axis shears in device space. In radians.

#### Parameter

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

#### Return

The angle that text space Y-axis shears in device space. In radians.

#### Head file reference

fpd\_pageobjTempl.h: 562

## FPDTextStateGetTextCTM

#### Syntax

```
void FPDTextStateGetTextCTM (
    FPD\_TextState textState,
    FS\_FLOAT* outCTM
);
```

#### Description

Gets the CTM for stroking purpose. The CTM array must be allocated and freed by caller.  
It must contain 4 elements.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

outCTM	[Out] It receives the text CTM for stroking purpose.
--------	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 640

**FPDTextStateGetTextMode****Syntax**

```
FS_INT32 FPDTextStateGetTextMode (
    FPD\_TextState textState
);
```

**Description**

Gets the text mode.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

**Return**

The text mode.

**Head file reference**

fpd\_pageobjTempl.h: 631

**FPDTextStateGetWordSpace****Syntax**

```
FS_FLOAT FPDTextStateGetWordSpace (
    FPD\_TextState textState
);
```

**Description**

Gets the word space.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

**Return**

The word space.

**Head file reference**

fpd\_pageobjTempl.h: 660

**FPDTextStateIsNull****Syntax**

```
FS_BOOL FPDTextStateIsNull (
    FPD\_TextState textState
);
```

**Description**

Tests whether the text state object is [NULL](#) or not.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

**Return**

Non-zero means [NULL](#) , otherwise not [NULL](#) .

**Head file reference**

fpd\_pageobjTempl.h: 669

**FPDTextStateNew****Syntax**

```
FPD_TextState FPDTextStateNew (void );
```

**Description**

Creates a new empty PDF text state object.

**Return**

A new empty PDF text state object.

**Head file reference**

fpd\_pageobjTempl.h: 480

**FPDTextStateSetCharSpace****Syntax**

```
void FPDTextStateSetCharSpace (
    FPD\_TextState textState,
    FS\_FLOAT fCharSpace
```

);

**Description**

Sets the character space.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

fCharSpace	[In] The input character space.
------------	---------------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 581

**FPDTextStateSetFont****Syntax**

```
void FPDTextStateSetFont (
    FPD\_TextState textState,
    FPD\_Font font
);
```

**Description**

Sets the font.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

font	[In] The font to set.
------	-----------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 507

**FPDTextStateSetFontSize****Syntax**

```
void FPDTextStateSetFontSize (
    FPD\_TextState textState,
```

```
FS_FLOAT fontSize  
);
```

**Description**

Sets the font size.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

fontSize	[In] The input font size.
----------	---------------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 571

**FPDTextStateSetMatrix****Syntax**

```
void FPDTextStateSetMatrix (  
    FPD_TextState textState,  
    matrix  
);
```

**Description**

Sets the text transformation matrix.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

matrix	[In] The input text transformation matrix.
--------	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 601

**FPDTextStateSetTextCTM****Syntax**

```
void FPDTextStateSetTextCTM (
```

```
FPD_TextState textState,  
CTM  
);
```

**Description**

Sets the CTM for stroking purpose.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

CTM	[In] The input CTM.
-----	---------------------

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 621

**FPDTextStateSetTextMode****Syntax**

```
void FPDTextStateSetTextMode (  
    FPD_TextState textState,  
    FS_INT32 iTextMode  
) ;
```

**Description**

Sets the text mode.

**Parameter**

---

textState	[In] The input PDF text state object.
-----------	---------------------------------------

---

iTextMode	[In] The input text mode. Only 0 is valid in this version.
-----------	--

---

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 611

**FPDTextStateSetWordSpace****Syntax**

---

```
void FPDTTextStateSetWordSpace (
    FPD_TextState textState,
    FS_FLOAT fWordSpace
);
```

**Description**

Sets the word space.

**Parameter**

textState	[In] The input PDF text state object.
fWordSpace	[In] The input word space.

**Return**

void

**Head file reference**

fpd\_pageobjTempl.h: 591

## FPD\_TilingPattern

[Return from Used by](#)

**Description**

The tiling pattern. see *CHAPTER 4.6.2 in PDF reference*. Consist of colored tiling pattern and uncolored tiling pattern. A colored tiling pattern is a pattern whose color is self-contained. An uncolored tiling pattern is a pattern that has no inherent color. See [FPDTilingPatternNew](#) , [FPDTilingPatternDestroy](#) .

**Returned from**

[FPDTilingPatternNew](#)  
[FPDTilingPatternNewII](#)

**Used by**

[FPDTilingPatternDestroy](#)  
[FPDTilingPatternGetBBox](#)  
[FPDTilingPatternGetForm](#)  
[FPDTilingPatternGetPatternMatrix](#)  
[FPDTilingPatternGetPatternObj](#)  
[FPDTilingPatternGetPatternType](#)  
[FPDTilingPatternGetPDDoc](#)  
[FPDTilingPatternGetXStep](#)  
[FPDTilingPatternGetYStep](#)  
[FPDTilingPatternIsColored](#)  
[FPDTilingPatternLoad](#)

## Functions

### Functions summary

#### [FPDTilingPatternDestroy](#)

Destroys the PDF tiling pattern object.

#### [FPDTilingPatternGetBBox](#)

Gets the bounding box in pattern space.

#### [FPDTilingPatternGetForm](#)

Gets all objects contained in this pattern.

#### [FPDTilingPatternGetPatternMatrix](#)

Gets matrix from pattern to parent stream.

#### [FPDTilingPatternGetPatternObj](#)

Gets dictionary for shading, stream for tiling.

#### [FPDTilingPatternGetPatternType](#)

Gets the pattern type.

#### [FPDTilingPatternGetPDDoc](#)

Gets the PDF document.

#### [FPDTilingPatternGetXStep](#)

Gets the desired horizontal spacing between pattern cells. In pattern space, absolute values only.

#### [FPDTilingPatternGetYStep](#)

Gets the desired vertical spacing between pattern cells. In pattern space, absolute values only.

#### [FPDTilingPatternIsColored](#)

Tests whether the pattern is a colored pattern or not.

#### [FPDTilingPatternLoad](#)

Loads all following data.

#### [FPDTilingPatternNew](#)

Creates a new tiling pattern object.

#### [FPDTilingPatternNewII](#)

Creates a new tiling pattern object.

### Functions detail

#### FPDTilingPatternDestroy

##### Syntax

```
void FPDTilingPatternDestroy (
    FPD\_TilingPattern pattern
);
```

##### Description

Destroys the PDF tiling pattern object.

##### Parameter

pattern	[In] The input PDF tiling pattern object.
---------	---

##### Return

void

**Head file reference**

fpd\_resourceTempl.h: 1636

**FPDTilingPatternGetBBox****Syntax**

```
FS_FloatRect FPDTilingPatternGetBBox (
    FPD\_TilingPattern pattern
);
```

**Description**

Gets the bounding box in pattern space.

**Parameter**

---

pattern	[In] The input PDF tiling pattern object.
---------	---

---

**Return**

The bounding box. In pattern space.

**Head file reference**

fpd\_resourceTempl.h: 1699

**FPDTilingPatternGetForm****Syntax**

```
FPD_Form FPDTilingPatternGetForm (
    FPD\_TilingPattern pattern
);
```

**Description**

Gets all objects contained in this pattern.

**Parameter**

---

pattern	[In] The input PDF tiling pattern object.
---------	---

---

**Return**

All objects contained in this pattern. [NULL](#) if not loaded.

**Head file reference**

fpd\_resourceTempl.h: 1726

**FPDTilingPatternGetPatternMatrix**

**Syntax**

```
FS_AffineMatrix FPDTilingPatternGetPatternMatrix (
    FPD\_TilingPattern pattern
);
```

**Description**

Gets matrix from pattern to parent stream.

**Parameter**

---

pattern	[In] The input PDF tiling pattern object.
---------	---

---

**Return**

The matrix from pattern to parent stream.

**Head file reference**

fpd\_resourceTempl.h: 1663

**FPDTilingPatternGetPatternObj****Syntax**

```
FPD_Object FPDTilingPatternGetPatternObj (
    FPD\_TilingPattern pattern
);
```

**Description**

Gets dictionary for shading, stream for tiling.

**Parameter**

---

pattern	[In] The input PDF tiling pattern object.
---------	---

---

**Return**

A dictionary for shading, stream for tiling.

**Head file reference**

fpd\_resourceTempl.h: 1645

**FPDTilingPatternGetPatternType****Syntax**

```
FS_INT32 FPDTilingPatternGetPatternType (
    FPD\_TilingPattern pattern
);
```

**Description**

Gets the pattern type.

#### Parameter

---

pattern	[In] The input PDF tiling pattern object.
---------	---

---

#### Return

The pattern type.

#### Head file reference

fpd\_resourceTempl.h: 1654

### FPDTilingPatternGetPDDoc

#### Syntax

```
FPD_Document FPDTilingPatternGetPDDoc (
    FPD\_TilingPattern pattern
);
```

#### Description

Gets the PDF document.

#### Parameter

---

pattern	[In] The input PDF tiling pattern object.
---------	---

---

#### Return

The PDF document associated with the tiling pattern.

#### Head file reference

fpd\_resourceTempl.h: 1672

### FPDTilingPatternGetXStep

#### Syntax

```
FS_FLOAT FPDTilingPatternGetXStep (
    FPD\_TilingPattern pattern
);
```

#### Description

Gets the desired horizontal spacing between pattern cells. In pattern space, absolute values only.

#### Parameter

---

pattern	[In] The input PDF tiling pattern object.
---------	---

---



**Return**

The desired horizontal spacing between pattern cells.

**Head file reference**

fpd\_resourceTempl.h: 1708

**FPDTilingPatternGetYStep****Syntax**

```
FS_FLOAT FPDTilingPatternGetYStep (
    FPD\_TilingPattern pattern
);
```

**Description**

Gets the desired vertical spacing between pattern cells. In pattern space, absolute values only.

**Parameter**

---

pattern	[In] The input PDF tiling pattern object.
---------	---

---

**Return**

The desired vertical spacing between pattern cells.

**Head file reference**

fpd\_resourceTempl.h: 1711

**FPDTilingPatternIsColored****Syntax**

```
FS_BOOL FPDTilingPatternIsColored (
    FPD\_TilingPattern pattern
);
```

**Description**

Tests whether the pattern is a colored pattern or not.

**Parameter**

---

pattern	[In] The input PDF tiling pattern object.
---------	---

---

**Return**

[TRUE](#) if the pattern is a colored pattern, otherwise [FALSE](#).

**Head file reference**

fpd\_resourceTempl.h: 1690



## FPDTilingPatternLoad

### Syntax

```
FS_BOOL FPDTilingPatternLoad (
    FPD\_TilingPattern pattern
);
```

### Description

Loads all following data.

### Parameter

---

pattern	[In] The input PDF tiling pattern object.
---------	---

---

### Return

[TRUE](#) if the data is loaded, otherwise [FALSE](#).

### Head file reference

fpd\_resourceTempl.h: 1681

## FPDTilingPatternNew

### Syntax

```
FPD_TilingPattern FPDTilingPatternNew (
    FPD\_Document doc,
    FPD\_Object obj
);
```

### Description

Creates a new tiling pattern object.

### Parameter

---

doc	[In] The PDF document.
-----	------------------------

---

---

obj	[In] The PDF object of the pattern.
-----	-------------------------------------

---

### Return

A new tiling pattern object.

### Head file reference

fpd\_resourceTempl.h: 1626

**Note:** Not support since SDK\_LATEEST\_VERSION > 1.0. You can use FPDTilingPatternNewII instead.

## FPDTilingPatternNewII

### Syntax

```
FPD_TilingPattern FPDTilingPatternNewII (
    FPD Document doc,
    FPD Object obj,
    FS AffineMatrix parentMatrix
);
```

### Description

Creates a new tiling pattern object.

### Parameter

---

doc	[In] The PDF document.
-----	------------------------

---

obj	[In] The PDF object of the pattern.
-----	-------------------------------------

---

parentMatrix	[In] The parent matrix.
--------------	-------------------------

---

### Return

A new tiling pattern object.

### Head file reference

fpd\_resourceTempl.h: 1631

### Since

[SDK LATEEST VERSION > 1.0](#)

## FPD\_TrueTypeFont

### Description

The True-Type font.

### Functions

#### Functions summary

##### [\*\*FPDTrueTypeFontDestroy\*\*](#)

Destroys the True-type font.

##### [\*\*FPDTrueTypeFontGetCharBBox\*\*](#)

Gets the bounding box of a character.

##### [\*\*FPDTrueTypeFontGetCharWidthF\*\*](#)

The PDF width of the character.

**[FPDTrueTypeFontGetEncoding](#)**

Gets the font encoding.

**[FPDTrueTypeFontGlyphFromCharCode](#)**

Gets the glyph index for a charcode.

**[FPDTrueTypeFontIsUnicodeCompatible](#)**

Tests whether a True-type font is compatible for unicode.

**[FPDTrueTypeFontNew](#)**

Creates a new True-type font.

## Functions detail

**FPDTrueTypeFontDestroy****Syntax**

```
void FPDTrueTypeFontDestroy (
    FPD\_Font font
);
```

**Description**

Destroys the True-type font.

**Parameter**

---

font	[In] The input True-type font.
------	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 744

**FPDTrueTypeFontGetCharBBox****Syntax**

```
FS_Rect FPDTrueTypeFontGetCharBBox (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

**Description**

Gets the bounding box of a character.

**Parameter**

---

font	[In] The input True-type font.
------	--------------------------------

---

charcode	[In] Input a charcode.
----------	------------------------

---

**Return**

The bounding box of a character.

**Head file reference**

fpd\_resourceTempl.h: 772

**FPDTrueTypeFontGetCharWidthF****Syntax**

```
FS_INT32 FPDTrueTypeFontGetCharWidthF (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

**Description**

The PDF width of the character.

**Parameter**

---

font	[In] The input True-type font.
------	--------------------------------

---

charcode	[In] Input a charcode.
----------	------------------------

---

**Return**

The PDF width of the character.

**Head file reference**

fpd\_resourceTempl.h: 762

**FPDTrueTypeFontGetEncoding****Syntax**

```
FPD_FontEncoding FPDTrueTypeFontGetEncoding (
    FPD\_Font font
);
```

**Description**

Gets the font encoding.

**Parameter**

---

font	[In] The input True-type font.
------	--------------------------------

---

**Return**

The font encoding.

**Head file reference**

fpd\_resourceTempl.h: 753

**FPDTrueTypeFontGlyphFromCharCode****Syntax**

```
FS_INT32 FPDTrueTypeFontGlyphFromCharCode (
    FPD_Font font,
    FS_DWORD charcode
);
```

**Description**

Gets the glyph index for a charcode.

**Parameter**

---

font	[In] The input True-type font.
------	--------------------------------

---

charcode	[In] Input a charcode.
----------	------------------------

---

**Return**

The glyph index of the charcode. Return -1 for unknown.

**Head file reference**

fpd\_resourceTempl.h: 782

**Note:** For embedded font only.

**FPDTrueTypeFontIsUnicodeCompatible****Syntax**

```
FS_BOOL FPDTrueTypeFontIsUnicodeCompatible (
    FPD_Font font
);
```

**Description**

Tests whether a True-type font is compatible for unicode.

**Parameter**

---

font	[In] The input True-type font.
------	--------------------------------

---

**Return**

TRUE if the font is compatible for unicode, otherwise FALSE.

**Head file reference**

fpd\_resourceTempl.h: 792

## FPDTrueTypeFontNew

### Syntax

FPD\_Font FPDTrueTypeFontNew (void );

### Description

Creates a new True-type font.

### Return

A new empty PDF True-type font.

### Head file reference

fpd\_resourceTempl.h: 735

# FPD\_Type1Font

## Description

The Type1 font.

## Functions

### Functions summary

#### [FPDType1FontDestroy](#)

Destroys the PDF type1 font.

#### [FPDType1FontGetBase14Font](#)

Gets an ID of standard 14 font.

#### [FPDType1FontGetCharBBox](#)

Gets the bounding box of a character.

#### [FPDType1FontGetCharWidthF](#)

The PDF width of the character.

#### [FPDType1FontGetEncoding](#)

Gets the font encoding.

#### [FPDType1FontGlyphFromCharCode](#)

Gets the glyph index for a charcode.

#### [FPDType1FontIsUnicodeCompatible](#)

Tests whether the Type1 font is compatible for unicode.

#### [FPDType1FontNew](#)

Creates a new empty PDF type1 font.

## Functions detail

### [FPDType1FontDestroy](#)

**Syntax**

```
void FPDTyp1FontDestroy (
    FPD Font font
);
```

**Description**

Destroys the PDF type1 font.

**Parameter**

---

font	[In] The input PDF type1 font.
------	--------------------------------

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 661

**FPDTyp1FontGetBase14Font****Syntax**

```
FS_INT32 FPDTyp1FontGetBase14Font (
    FPD Font font
);
```

**Description**

Gets an ID of standard 14 font.

**Parameter**

---

font	[In] The input PDF type1 font.
------	--------------------------------

---

**Return**

An ID of standard 14 font. -1 for none. See list of standard fonts in PDF Reference.

**Head file reference**

fpd\_resourceTempl.h: 718

**FPDTyp1FontGetCharBBox****Syntax**

```
FS_Rect FPDTyp1FontGetCharBBox (
    FPD Font font,
    FS\_DWORD charcode
);
```

**Description**

Gets the bounding box of a character.

**Parameter**

---

font	[In] The input PDF type1 font.
------	--------------------------------

---

charcode	[In] Input a charcode.
----------	------------------------

---

**Return**

The bounding box of a character.

**Head file reference**

fpd\_resourceTempl.h: 689

**FPDType1FontGetCharWidthF****Syntax**

```
FS_INT32 FPDType1FontGetCharWidthF (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

**Description**

The PDF width of the character.

**Parameter**

---

font	[In] The input PDF type1 font.
------	--------------------------------

---

charcode	[In] Input a charcode.
----------	------------------------

---

**Return**

The PDF width of the character.

**Head file reference**

fpd\_resourceTempl.h: 679

**FPDType1FontGetEncoding****Syntax**

```
FPD_FontEncoding FPDType1FontGetEncoding (
    FPD\_Font font
);
```



**Description**

Gets the font encoding.

**Parameter**

---

font	[In] The input PDF type1 font.
------	--------------------------------

---

**Return**

The font encoding.

**Head file reference**

fpd\_resourceTempl.h: 670

**FPDType1FontGlyphFromCharCode****Syntax**

```
FS_INT32 FPDType1FontGlyphFromCharCode (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

**Description**

Gets the glyph index for a charcode.

**Parameter**

---

font	[In] The input PDF type1 font.
------	--------------------------------

---

charcode	[In] Input a charcode.
----------	------------------------

---

**Return**

The glyph index of the charcode. Return -1 for unknown.

**Head file reference**

fpd\_resourceTempl.h: 699

**Note:** For embedded font only.

**FPDType1FontIsUnicodeCompatible****Syntax**

```
FS_BOOL FPDType1FontIsUnicodeCompatible (
    FPD\_Font font
);
```

**Description**

Tests whether the Type1 font is compatible for unicode.

#### Parameter

font	[In] The input PDF type1 font.
------	--------------------------------

#### Return

[TRUE](#) if the font is compatible for unicode, otherwise [FALSE](#).

#### Head file reference

fpd\_resourceTempl.h: 709

## FPDType1FontNew

#### Syntax

```
FPD_Font FPDType1FontNew (void );
```

#### Description

Creates a new empty PDF type1 font.

#### Return

A new empty PDF type1 font.

#### Head file reference

fpd\_resourceTempl.h: 652

## FPD\_Type3Char

### [Return from Used by](#)

#### Description

Type3 character information. See [FPDType3CharNew](#) , [FPDType3CharDestroy](#) .

#### Returned from

[FPDType3FontLoadChar](#)  
[FPDType3CharNew](#)

#### Used by

[FPDFFormParseContent](#)  
[FPDFFormStartParse](#)  
[FPDType3CharDestroy](#)  
[FPDType3CharGetBBox](#)  
[FPDType3CharGetDIBitmap](#)  
[FPDType3CharGetForm](#)

---

[\*\*FPDType3CharGetImageMatrix\*\*](#)  
[\*\*FPDType3CharGetWidth\*\*](#)  
[\*\*FPDType3CharIsColored\*\*](#)  
[\*\*FPDType3CharIsPageRequired\*\*](#)

## Functions

### Functions summary

[\*\*FPDType3CharDestroy\*\*](#)

Destroys the type3 character information object.

[\*\*FPDType3CharGetBBox\*\*](#)

Gets the character's bounding box. In font coordinate (1/1000 of em).

[\*\*FPDType3CharGetDIBitmap\*\*](#)

Gets the image pointer if it's a image.

[\*\*FPDType3CharGetForm\*\*](#)

Gets the form pointer if it's a form.

[\*\*FPDType3CharGetImageMatrix\*\*](#)

Gets the image matrix if it's a image

[\*\*FPDType3CharGetWidth\*\*](#)

Gets the character width. In font coordinate (1/1000 of em).

[\*\*FPDType3CharIsColored\*\*](#)

Tests whether the type3 character is colored.

[\*\*FPDType3CharIsPageRequired\*\*](#)

Tests whether the Type3 font is using a page resource.

[\*\*FPDType3CharNew\*\*](#)

Creates a new empty type3 character information object.

### Functions detail

[\*\*FPDType3CharDestroy\*\*](#)

**Syntax**

```
void FPDType3CharDestroy (
    FPD\_Type3Char type3char
);
```

**Description**

Destroys the type3 character information object.

**Parameter**

type3char	[In] The input type3 character information object.
-----------	--

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 818

## FPDType3CharGetBBox

### Syntax

```
FS_Rect FPDType3CharGetBBox (
    FPD\_Type3Char type3char
);
```

### Description

Gets the character's bounding box. In font coordinate (1/1000 of em).

### Parameter

---

type3char	[In] The input type3 character information object.
-----------	--

---

### Return

The character's bounding box. In font coordinate (1/1000 of em).

### Head file reference

fpd\_resourceTempl.h: 881

## FPDType3CharGetDIBitmap

### Syntax

```
FS_DIBitmap FPDType3CharGetDIBitmap (
    FPD\_Type3Char type3char
);
```

### Description

Gets the image pointer if it's a image.

### Parameter

---

type3char	[In] The input type3 character information object.
-----------	--

---

### Return

The image pointer if it's a image.

### Head file reference

fpd\_resourceTempl.h: 863

## FPDType3CharGetForm

### Syntax

```
FPD_Form FPDType3CharGetForm (
    FPD\_Type3Char type3char
);
```

**Description**

Gets the form pointer if it's a form.

**Parameter**

---

type3char	[In] The input type3 character information object.
-----------	--

---

**Return**

The form pointer if it's a form.

**Head file reference**

fpd\_resourceTempl.h: 845

**FPDType3CharGetImageMatrix****Syntax**

```
FS_AffineMatrix FPDType3CharGetImageMatrix (
    FPD\_Type3Char type3char
);
```

**Description**

Gets the image matrix if it's a image

**Parameter**

---

type3char	[In] The input type3 character information object.
-----------	--

---

**Return**

The image matrix if it's a image

**Head file reference**

fpd\_resourceTempl.h: 854

**FPDType3CharGetWidth****Syntax**

```
FS_INT32 FPDType3CharGetWidth (
    FPD\_Type3Char type3char
);
```

**Description**

Gets the character width. In font coordinate (1/1000 of em).

**Parameter**

---

type3char	[In] The input type3 character information object.
-----------	--

---

**Return**

The character width. In font coordinate (1/1000 of em).

**Head file reference**

fpd\_resourceTempl.h: 872

**FPDType3CharIsColored****Syntax**

```
FS_BOOL FPDType3CharIsColored (
    FPD\_Type3Char type3char
);
```

**Description**

Tests whether the type3 character is colored.

**Parameter**

---

type3char	[In] The input type3 character information object.
-----------	--

---

**Return**

[TRUE](#) if the type3 character is colored, otherwise [FALSE](#).

**Head file reference**

fpd\_resourceTempl.h: 827

**FPDType3CharIsPageRequired****Syntax**

```
FS_BOOL FPDType3CharIsPageRequired (
    FPD\_Type3Char type3char
);
```

**Description**

Tests whether the Type3 font is using a page resource.

**Parameter**

---

type3char	[In] The input type3 character information object.
-----------	--

---

**Return**

[TRUE](#) if using a page resource therefore we can't load the char.

**Head file reference**

fpd\_resourceTempl.h: 836

### FPDType3CharNew

#### Syntax

```
FPD_Type3Char FPDType3CharNew (void );
```

#### Description

Creates a new empty type3 character information object.

#### Return

A new empty type3 character information object.

#### Head file reference

fpd\_resourceTempl.h: 809

## FPD\_Type3Font

### Description

The Type3 font. Type 3 fonts do not have the ability to provide a base font with more than one encoding. For each Type 3 font, there is only one encoding. This encoding is completely specified in the PDF file; there are no shortcuts as there are for other fonts. See *Section 5.7, Font Descriptors, in the PDF Reference* for a discussion of font descriptors.

### Functions

#### Functions summary

##### [FPDType3FontDestroy](#)

Destroys the Type3 font.

##### [FPDType3FontGetCharBBox](#)

Gets the bounding box of a character.

##### [FPDType3FontGetCharTypeWidth](#)

Gets the PDF width of the character.

##### [FPDType3FontGetCharWidthF](#)

Gets the PDF width of the character.

##### [FPDType3FontGetEncoding](#)

Gets the font encoding.

##### [FPDType3FontGetFontMatrix](#)

Gets the matrix of the font.

##### [FPDType3FontGlyphFromCharCode](#)

Gets the glyph index for a charcode.

##### [FPDType3FontIsUnicodeCompatible](#)

Tests whether a Type3 font is compatible for unicode.

##### [FPDType3FontLoadChar](#)

Loads a character.

##### [FPDType3FontNew](#)

Creates a new empty Type3 font.

### [FPDType3FontSetPageResources](#)

Sets Page resources.

## Functions detail

### FPDType3FontDestroy

#### Syntax

```
void FPDType3FontDestroy (
    FPD\_Font font
);
```

#### Description

Destroys the Type3 font.

#### Parameter

---

font	[In] The input Type3 font.
------	----------------------------

---

#### Return

void

#### Head file reference

fpd\_resourceTempl.h: 907

### FPDType3FontGetCharBBox

#### Syntax

```
FS_Rect FPDType3FontGetCharBBox (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

#### Description

Gets the bounding box of a character.

#### Parameter

---

font	[In] The input Type3 font.
------	----------------------------

---

---

charcode	[In] Input a charcode.
----------	------------------------

---

#### Return

The bounding box of a character.

#### Head file reference



fpd\_resourceTempl.h: 935

## FPDType3FontGetCharTypeWidth

### Syntax

```
FS_INT32 FPDType3FontGetCharTypeWidth (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

### Description

Gets the PDF width of the character.

### Parameter

---

font	[In] The input Type3 font.
------	----------------------------

---

charcode	[In] The input character code.
----------	--------------------------------

---

### Return

the PDF width of the character.

### Head file reference

fpd\_resourceTempl.h: 984

## FPDType3FontGetCharWidthF

### Syntax

```
FS_INT32 FPDType3FontGetCharWidthF (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

### Description

Gets the PDF width of the character.

### Parameter

---

font	[In] The input Type3 font.
------	----------------------------

---

charcode	[In] Input a charcode.
----------	------------------------

---

### Return

The PDF width of the character.



**Head file reference**

fpd\_resourceTempl.h: 925

**FPDType3FontGetEncoding****Syntax**

```
FPD_FontEncoding FPDType3FontGetEncoding (
    FPD_Font font
);
```

**Description**

Gets the font encoding.

**Parameter**

---

font	[In] The input Type3 font.
------	----------------------------

---

**Return**

The font encoding.

**Head file reference**

fpd\_resourceTempl.h: 916

**FPDType3FontGetFontMatrix****Syntax**

```
FS_AffineMatrix FPDType3FontGetFontMatrix (
    FPD_Font font
);
```

**Description**

Gets the matrix of the font.

**Parameter**

---

font	[In] The input Type3 font.
------	----------------------------

---

**Return**

The matrix of the font.

**Head file reference**

fpd\_resourceTempl.h: 994

**FPDType3FontGlyphFromCharCode****Syntax**

```
FS_INT32 FPDType3FontGlyphFromCharCode (
    FPD\_Font font,
    FS\_DWORD charcode
);
```

**Description**

Gets the glyph index for a charcode.

**Parameter**

---

font	[In] The input Type3 font.
------	----------------------------

---

charcode	[In] Input a charcode.
----------	------------------------

---

**Return**

The glyph index of the charcode. Return -1 for unknown.

**Head file reference**

fpd\_resourceTempl.h: 945

**Note:** For embedded font only.

**FPDType3FontIsUnicodeCompatible****Syntax**

```
FS_BOOL FPDType3FontIsUnicodeCompatible (
    FPD\_Font font
);
```

**Description**

Tests whether a Type3 font is compatible for unicode.

**Parameter**

---

font	[In] The input Type3 font.
------	----------------------------

---

**Return**

[TRUE](#) if font is compatible for unicode, otherwise [FALSE](#).

**Head file reference**

fpd\_resourceTempl.h: 955

**FPDType3FontLoadChar****Syntax**

```
FPD_Type3Char FPDType3FontLoadChar (
```



```
FPD_Font font,  
FS_DWORD charcode  
);
```

**Description**

Loads a character.

**Parameter**

---

font	[In] The input Type3 font.
------	----------------------------

---

charcode	[In] Input a charcode.
----------	------------------------

---

**Return**

The type3 character information.

**Head file reference**

fpd\_resourceTempl.h: 974

**FPDType3FontNew****Syntax**

```
FPD_Font FPDType3FontNew (void );
```

**Description**

Creates a new empty Type3 font.

**Return**

A new empty Type3 font.

**Head file reference**

fpd\_resourceTempl.h: 898

**FPDType3FontSetPageResources****Syntax**

```
void FPDType3FontSetPageResources (  
    FPD_Font font,  
    FPD_Object resourcesDic  
) ;
```

**Description**

Sets Page resources.

**Parameter**

---

font	[In] The input Type3 font.
resourcesDic	[In] the Dictionary contenting the resource.

---

**Return**

void

**Head file reference**

fpd\_resourceTempl.h: 964

## FPD\_UnencryptedWrapperCreator

**[Return from Used by](#)**

### Description

#### Returned from

**[FPDUnencryptedWrapperCreatorNew](#)**

#### Used by

**[FPDUnencryptedWrapperCreatorDestroy](#)****[FPDUnencryptedWrapperCreatorSetPayloadInfo](#)****[FPDUnencryptedWrapperCreatorSetPayLoad](#)****[FPDUnencryptedWrapperCreatorCreate](#)****[FPDUnencryptedWrapperCreatorContinue](#)****[FPDUnencryptedWrapperCreatorSetStandardSecurity](#)**

### Functions

#### Functions summary

**[FPDUnencryptedWrapperCreatorNew](#)**

Creates an instance of FPD\_Wrapper20Creator.

**[FPDUnencryptedWrapperCreatorDestroy](#)**

Destroy the wrapper creator.

**[FPDUnencryptedWrapperCreatorSetPayloadInfo](#)**

Set data of the embedded encrypted payload document for the wrapper doc.

**[FPDUnencryptedWrapperCreatorSetPayLoad](#)**

Set the embedded encrypted payload document for the wrapper doc.

**[FPDUnencryptedWrapperCreatorCreate](#)**

Write the wrapper version to a custom file access.

**[FPDUnencryptedWrapperCreatorContinue](#)**

Continue to write wrapper document data under progressive mode.

**[FPDUnencryptedWrapperCreatorSetStandardSecurity](#)**

Set security settings using standard security handler only.

## Functions detail

### FPDUnencryptedWrapperCreatorNew

#### Syntax

```
FPD_UnencryptedWrapperCreator FPDUnencryptedWrapperCreatorNew (
    FPD Document pWrapperDoc
);
```

#### Description

Creates an instance of FPD\_Wrapper20Creator.

#### Parameter

---

pWrapperDoc	[In] A document object which defines wrapper version, caller maintains its life-time.
-------------	---

---

#### Return

An instance of interface FPD\_Wrapper20Creator, NULL pointer if error happens.

#### Head file reference

fpd\_serialTempl.h: 790

#### Since

[SDK LATEEST VERSION > 8.3.1](#)

### FPDUnencryptedWrapperCreatorDestroy

#### Syntax

```
void FPDUnencryptedWrapperCreatorDestroy (
    FPD UnencryptedWrapperCreator wrapperCreator
);
```

#### Description

Destroy the wrapper creator.

#### Parameter

---

wrapperCreator	[In] The input wrapper creator object.
----------------	--

---

#### Return

void.

#### Head file reference

fpd\_serialTempl.h: 800

#### Since

[SDK LATEEST VERSION > 8.3.1](#)

**FPDUnencryptedWrapperCreatorSetPayloadInfo****Syntax**

```
void FPDUnencryptedWrapperCreatorSetPayloadInfo (
    FPD_UnencryptedWrapperCreator wrapperCreator,
    FS_LPWSTR wsSubType,
    FS_LPWSTR wsFileName,
    FS_LPWSTR wsDescription,
    FS_FLOAT fVersion
);
```

**Description**

Set data of the embedded encrypted payload document for the wrapper doc.

**Parameter**

wrapperCreator	[In] The input wrapper creator object.
wsSubType	[In] The name of the cryptographic filter used to encrypt the encrypted payload document.
wsFileName	[In] The file name for encrypted payload document which shall include the name of the cryptographic filter needed to decrypt the document.
wsDescription	[In] Description text for the embedded encrypted payload document.
fVersion	[In] The version number of the cryptographic filter used to encrypt the encrypted payload.

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 810

**Note:** The Param "bsFileName" must not contain or be derived from the encrypted payloadâ€™s actual file name. This is to avoid potential disclosure of sensitive information in the original filename.

**Since**

[SDK LATEEST VERSION > 8.3.1](#)

**FPDUnencryptedWrapperCreatorSetPayLoad**

**Syntax**

```
void FPDUnencryptedWrapperCreatorSetPayLoad (
    FPD UnencryptedWrapperCreator wrapperCreator
);
```

**Description**

Set the embedded encrypted payload document for the wrapper doc.

**Parameter**


---

wrapperCreator	[In] The input wrapper creator object. param[in]: pPayload The embedded encrypted payload document.
----------------	---

---

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 826

**Since**

[SDK\\_LATEEST\\_VERSION > 8.3.1](#)

**FPDUnencryptedWrapperCreatorCreate****Syntax**

```
FS_BOOL FPDUnencryptedWrapperCreatorCreate (
    FPD UnencryptedWrapperCreator wrapperCreator,
    pFile,
    FS\_DWORD flags
);
```

**Description**

Write the wrapper version to a custom file access.

**Parameter**


---

wrapperCreator	[In] The input wrapper creator object.
----------------	--

---

pFile	[Out] The output file access.
-------	-------------------------------

---

flags	[In] The creating flags.
-------	--------------------------

---

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_serialTempl.h: 837



**Since**[SDK LATEEST VERSION > 8.3.1](#)

## FPDUnencryptedWrapperCreatorContinue

**Syntax**

```
FS_INT32 FPDUnencryptedWrapperCreatorContinue (
    FPD\_UnencryptedWrapperCreator wrapperCreator,
    FS\_PauseHandler pPause
);
```

**Description**

Continue to write wrapper document data under progressive mode.

**Parameter**

wrapperCreator	[In] The input wrapper creator object.
----------------	--

pPause	[In] Pause object, optional.
--------	------------------------------

**Return**

Negative value if failure, 0 if finishes, and positive value if need to be continued.

**Head file reference**

fpd\_serialTempl.h: 849

**Note:** Only valid if pass FPDFCREATE\_PROGRESSIVE flag in calling Create method.

**Since**[SDK LATEEST VERSION > 8.3.1](#)

## FPDUnencryptedWrapperCreatorSetStandardSecurity

**Syntax**

```
void FPDUnencryptedWrapperCreatorSetStandardSecurity (
    FPD\_UnencryptedWrapperCreator wrapperCreator,
    FS\_DWORD permissions,
    FS\_LPCBYTE owner_pass,
    FS\_INT32 owner_pass_len
);
```

**Description**

Set security settings using standard security handler only.

**Parameter**

---

wrapperCreator	[In] The input wrapper creator object.
permissions	[In] The user permissions.
owner_pass	[In] The owner password.
owner_pass_len	[In] The length of owner password.

---

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 860

**Note:** Can't be used with incremental update.

**Since**[SDK\\_LATEEST\\_VERSION > 9.1](#)

## FPD\_WindowsDevice

### Description

The rendering device based on Windows device driver.

### Functions

#### Functions summary

##### [FPDWindowsDeviceDestroy](#)

Destroys the PDF Windows device object.

##### [FPDWindowsDeviceNew](#)

Creates a new empty Windows device object.

#### Functions detail

##### [FPDWindowsDeviceDestroy](#)

###### **Syntax**

```
void FPDWindowsDeviceDestroy (
    FPD\_RenderDevice dc
);
```

###### **Description**

Destroys the PDF Windows device object.

**Parameter**

---

dc	[In] The input PDF Windows device object.
----	---

---

**Return**

void

**Head file reference**

fpd\_renderTempl.h: 873

**FPDWindowsDeviceNew****Syntax**

```
FPD_RenderDevice FPDWindowsDeviceNew (
    void* windowsDC
);
```

**Description**

Creates a new empty Windows device object.

**Parameter**

---

windowsDC	[In] The input Windows device.
-----------	--------------------------------

---

**Return**

A new empty Windows rendering device object.

**Head file reference**

fpd\_renderTempl.h: 864

**FPD\_Wrapper20Creator****Description**

PDF Wrapper Creator: creating a wrapper 2.0 PDF file to an existing PDF file. See [FPDWrapper20CreatorNew](#) , [FPDWrapper20CreatorDestroy](#) .

**FPD\_WrapperCreator****Return from Used by****Description**

PDF Wrapper Creator: creating a wrapper PDF file to an existing PDF file. See [FPDWrapperCreatorNew](#) , [FPDWrapperCreatorDestroy](#) .

**Returned from**[\*\*FPDWrapperCreatorNew\*\*](#)**Used by**

[\*\*FPDWrapperCreatorCreate\*\*](#)  
[\*\*FPDWrapperCreatorDestroy\*\*](#)  
[\*\*FPDWrapperCreatorSetStandardSecurity\*\*](#)  
[\*\*FPDWrapperCreatorSetWrapperData\*\*](#)

**Functions****Functions summary**[\*\*FPDWrapperCreatorCreate\*\*](#)

Write the wrapper version to a custom file access.

[\*\*FPDWrapperCreatorDestroy\*\*](#)

Destroy the wrapper creator.

[\*\*FPDWrapperCreatorNew\*\*](#)

Creates an instance of FPD\_WrapperCreator.

[\*\*FPDWrapperCreatorSetStandardSecurity\*\*](#)

Set security settings using standard security handler only. Can't be used with incremental update.

[\*\*FPDWrapperCreatorSetWrapperData\*\*](#)

Set wrapper data.

**Functions detail**[\*\*FPDWrapperCreatorCreate\*\*](#)**Syntax**

```
FS_BOOL FPDWrapperCreatorCreate (
    FPD\_WrapperCreator wrapperCreator,
    FS\_StreamWriteHandler* pFile
);
```

**Description**

Write the wrapper version to a custom file access.

**Parameter**

wrapperCreator	[In] The input wrapper creator object.
----------------	--

pFile	[In] The output file access.
-------	------------------------------

**Return**

Non-zero means success, otherwise failure.

**Head file reference**

fpd\_serialTempl.h: 750

**Since**  
[SDK LATEEST VERSION > 1.0](#)

### FPDWrapperCreatorDestroy

#### Syntax

```
void FPDWrapperCreatorDestroy (
    FPD\_WrapperCreator wrapperCreator
);
```

#### Description

Destroy the wrapper creator.

#### Parameter

---

wrapperCreator	[In] The input wrapper creator object.
----------------	--

---

#### Return

void.

#### Head file reference

fpd\_serialTempl.h: 725

#### Since

[SDK LATEEST VERSION > 1.0](#)

### FPDWrapperCreatorNew

#### Syntax

```
FPD_WrapperCreator FPDWrapperCreatorNew (
    FPD\_Document pWrapperDoc,
    FS\_DWORD dwWrapperOffset
);
```

#### Description

Creates an instance of FPD\_WrapperCreator.

#### Parameter

---

pWrapperDoc	[In] A document object which defines wrapper version, caller maintains its life-time.
-------------	---

---

dwWrapperOffset	[In] Offset in bytes from the beginning of PDF file, for wrapper version.
-----------------	---

---

**Return**

An instance of interface FPD\_WrapperCreator, NULL pointer if error happens.

**Head file reference**

fpd\_serialTempl.h: 714

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDWrapperCreatorSetStandardSecurity****Syntax**

```
void FPDWrapperCreatorSetStandardSecurity (
    FPD\_WrapperCreator wrapperCreator,
    FS\_DWORD permissions,
    FS\_LPCBYTE owner_pass,
    FS\_INT32 owner_pass_len
);
```

**Description**

Set security settings using standard security handler only. Can't be used with incremental update.

**Parameter**


---

wrapperCreator	[In] The input wrapper creator object.
----------------	--

---

permissions	[In] The user permissions.
-------------	----------------------------

---

owner_pass	[In] The owner password.
------------	--------------------------

---

owner_pass_len	[In] The length of owner password.
----------------	------------------------------------

---

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 761

**Since**

[SDK\\_LATEEST\\_VERSION > 1.0](#)

**FPDWrapperCreatorSetWrapperData****Syntax**

```
void FPDWrapperCreatorSetWrapperData (
```

---

```
FPD_WrapperCreator wrapperCreator,
FS_LPSTR szType,
FS_INT32 iVersion,
FS_LPSTR szApplication,
FS_LPSTR szURI,
FS_LPSTR szDescription
);
```

**Description**

Set wrapper data.

**Parameter**

wrapperCreator	[In] The input wrapper creator object.
szType	[In] Wrapper type name.
iVersion	[In] Wrapper version.
szApplication	[In] Application identity which indicates how to process the current wrapper file.
szURI	[In] An URI site to retrieve more helpful information if necessary.
szDescription	[In] Description text for application.

**Return**

void.

**Head file reference**

fpd\_serialTempl.h: 735

**Since**

[SDK LATEEST VERSION > 1.0](#)

## FPD\_WrapperDoc

[Return from Used by](#)

**Description****Returned from**

[FPDWrapperDocNew](#)



## Used by

[FPDWrapperDocContinue](#)  
[FPDWrapperDocDestroy](#)  
[FPDWrapperDocGetCryptographicFilter](#)  
[FPDWrapperDocGetPayloadFileName](#)  
[FPDWrapperDocGetPayLoadSize](#)  
[FPDWrapperDocGetWrapperType](#)  
[FPDWrapperDocStartGetPayload](#)

## Definitions

### Definitions summary

#### **FPD\_WRAPPERTYPE\_FOXIT**

For Foxit wrapper document.

#### **FPD\_WRAPPERTYPE\_NO**

Normal document.

#### **FPD\_WRAPPERTYPE\_PDF2**

For PDF2.0 wrapper document.

### Definitions detail

#### **FPD\_WRAPPERTYPE\_FOXIT**

##### **Syntax**

```
#define FPD_WRAPPERTYPE_FOXIT 1
```

##### **Description**

For Foxit wrapper document.

##### **Group**

[FPDWrapperDocGetWrapperType](#)

##### **Head file reference**

fpd\_docExpT.h: 1419

#### **FPD\_WRAPPERTYPE\_NO**

##### **Syntax**

```
#define FPD_WRAPPERTYPE_NO 0
```

##### **Description**

Normal document.

##### **Group**

[FPDWrapperDocGetWrapperType](#)

##### **Head file reference**

fpd\_docExpT.h: 1417

## FPD\_WRAPPERTYPE\_PDF2

### Syntax

```
#define FPD_WRAPPERTYPE_PDF2 2
```

### Description

For PDF2.0 wrapper document.

### Group

[FPDWrapperDocGetWrapperType](#)

### Head file reference

fpd\_docExpT.h: 1421

## Functions

### Functions summary

#### [FPDWrapperDocContinue](#)

Continue getting the embedded encrypted payload document's file stream.

#### [FPDWrapperDocDestroy](#)

Destroys the wrapper document object.

#### [FPDWrapperDocGetCryptographicFilter](#)

Get the details of the cryptographic filter needed to decrypt the encrypted payload.

#### [FPDWrapperDocGetPayloadFileName](#)

Get the file name for encrypted payload documents which shall include the name of the cryptographic filter needed to decrypt the document.

#### [FPDWrapperDocGetPayLoadSize](#)

Get the file size of the uncompressed encrypted payload document.

#### [FPDWrapperDocGetWrapperType](#)

Whether the document is a wrapper document or normal.

#### [FPDWrapperDocNew](#)

Create the wrapper document object.

#### [FPDWrapperDocStartGetPayload](#)

Get the embedded encrypted payload document's file stream.

### Functions detail

#### FPDWrapperDocContinue

##### Syntax

```
FS_INT32 FPDWrapperDocContinue (
    FPD\_WrapperDoc wrapperDoc,
    FS\_PauseHandler pPause
);
```

##### Description

Continue getting the embedded encrypted payload document's file stream.

**Parameter**

---

wrapperDoc	[In] The input wrapper document object.
------------	---

---

**pPause**

[In] The user-supplied pause object.

---

**Return**

Negative value if failure, 0 if finishes, and positive value if need to be continued.

**Head file reference**

fpd\_docTempl.h: 6677

**Since**

[SDK\\_LATEEST\\_VERSION > 8.3.1](#)

**FPDWrapperDocDestroy****Syntax**

```
void FPDWrapperDocDestroy (
    FPD\_WrapperDoc wrapperDoc
);
```

**Description**

Destroys the wrapper document object.

**Parameter**

---

wrapperDoc	[In] The input wrapper document object.
------------	---

---

**Return**

void.

**Head file reference**

fpd\_docTempl.h: 6607

**Since**

[SDK\\_LATEEST\\_VERSION > 8.3.1](#)

**FPDWrapperDocGetCryptographicFilter****Syntax**

```
FS_BOOL FPDWrapperDocGetCryptographicFilter (
    FPD\_WrapperDoc wrapperDoc,
    wsGraphicFilter,
    fVersion
);
```

**Description**

Get the details of the cryptographic filter needed to decrypt the encrypted payload.

**Parameter**

wrapperDoc	[In] The input wrapper document object.
wsGraphicFilter	[Out] The name of the cryptographic filter.
fVersion	[Out] The version number of the cryptographic filter, if present; Otherwise, 0.

**Return**

TRUE if success, otherwise return FALSE.

**Head file reference**

fpd\_docTempl.h: 6630

**Since**

[SDK LATEEST VERSION > 8.3.1](#)

**FPDWrapperDocGetPayloadFileName****Syntax**

```
FS_BOOL FPDWrapperDocGetPayloadFileName (
    FPD\_WrapperDoc wrapperDoc,
    wsFileName
);
```

**Description**

Get the file name for encrypted payload documents which shall include the name of the cryptographic filter needed to decrypt the document.

**Parameter**

wrapperDoc	[In] The input wrapper document object.
wsFileName	[Out] The file name for encrypted payload documents.

**Return**

TRUE if success, otherwise return FALSE.

**Head file reference**

fpd\_docTempl.h: 6652



**Note:** The name must not contain or be derived from the encrypted payload's actual file name. This is to avoid potential disclosure of sensitive information in the original filename.

**Since**

[SDK LATEST VERSION > 8.3.1](#)

## FPDWrapperDocGetPayLoadSize

**Syntax**

```
FS_INT64 FPDWrapperDocGetPayLoadSize (
    FPD\_WrapperDoc wrapperDoc
);
```

**Description**

Get the file size of the uncompressed encrypted payload document.

**Parameter**

---

wrapperDoc	[In] The input wrapper document object.
------------	---

---

**Return**

The size of payload;

**Head file reference**

fpd\_docTempl.h: 6642

**Note:** if no "size" key, return -1;

**Since**

[SDK LATEST VERSION > 8.3.1](#)

## FPDWrapperDocGetWrapperType

**Syntax**

```
FS_INT32 FPDWrapperDocGetWrapperType (
    FPD\_WrapperDoc wrapperDoc
);
```

**Description**

Whether the document is a wrapper document or normal.

**Parameter**

---

wrapperDoc	[In] The input wrapper document object.
------------	---

---

**Return**

-1 for error; [PDF\\_WRAPPERTYPE\\_NO](#) for no wrapper document;  
[PDF\\_WRAPPERTYPE\\_FOXIT](#) for Foxit wrapper document;  
[PDF\\_WRAPPERTYPE\\_PDF2](#) for PDF2.0 wrapper document.

**Head file reference**

fpd\_docTempl.h: 6617

**Since**

[SDK LATEEST VERSION > 8.3.1](#)

## FPDWrapperDocNew

**Syntax**

```
FPD_WrapperDoc FPDWrapperDocNew (
    FPD Document fpdDoc
);
```

**Description**

Create the wrapper document object.

**Parameter**

---

fpdDoc	[In] The input document.
--------	--------------------------

---

**Return**

The wrapper document object.

**Head file reference**

fpd\_docTempl.h: 6597

**Since**

[SDK LATEEST VERSION > 8.3.1](#)

## FPDWrapperDocStartGetPayload

**Syntax**

```
FS_BOOL FPDWrapperDocStartGetPayload (
    FPD\_WrapperDoc wrapperDoc,
    pPayload,
    FS\_PauseHandler pPause
);
```

**Description**

Get the embedded encrypted payload document's file stream.

**Parameter**

wrapperDoc	[In] The input wrapper document object.
pPayload	[Out] The embedded encrypted payload document's file stream.
pPause	[In] The user-supplied pause object.

**Return**

TRUE, to be continued, otherwise, finished.

**Head file reference**

fpd\_docTempl.h: 6665

**Since**

[SDK\\_LATEEST\\_VERSION > 8.3.1](#)

## FRMS\_Template

### Description

A [FRMS\\_Template](#) is an object that represents the rights policy template of RMS. A [FRMS\\_Template](#) is an object that represents the rights policy template of RMS. User can create the object from the official template file, and can also create an empty template object by invoking [FRMSTemplateNew2](#) ().

## Definition Groups

---

### AddHFTErrs

Flags returned by FSExtensionHFTMgrAddHFT.

### List

[ERR\\_ADDHFT\\_EMPTYLHFT](#)  
[ERR\\_ADDHFT\\_NAMEEXIST](#)  
[ERR\\_ADDHFT\\_SUCCESS](#)  
[ERR\\_ADDHFT\\_UNKNOWN](#)

### CALL\_REPLACE\_PROC

Macros for entries replacing.

### List



### **CALL REPLACED PRO**

#### **FSBoolean**

FSBoolean

List

**FALSE**  
**TRUE**

#### **FSBoolean**

FS\_DEFINEHANDLE

List

**FS\_DEFINEHANDL**

#### **FSFormattingFlags**

The formating flags.

List

**FS FORMAT CAPITAL**  
**FS FORMAT HEX**  
**FS FORMAT SIGNED**

#### **FSNull**

Macro definition for key NULL.

List

**NULL**

#### **FSUUIDType**

Macro definitions for UUID types.



## List

[FS\\_UIDTYPE\\_INVALID](#)  
[FS\\_UIDTYPE\\_WINDOWS\\_OTHER](#)  
[FS\\_UIDTYPE\\_WINDOWS\\_RANDOM](#)  
[FS\\_UIDTYPE\\_WINDOWS\\_RANDOM\\_HIGHQUALITY](#)  
[FS\\_UIDTYPE\\_WINDOWS\\_TIME\\_MAC](#)  
[FS\\_UIDTYPE\\_WINDOWS\\_TIME\\_RANDOM](#)

## Handshakeverssion

Known handshake versions and data structures should be listed below:

- Top 2 bytes contains the major version; bottom 2 bytes the minor version.
- All shipping versions have a zero minor version.
- All apps and plugs-ins must support all previous shipping versions.
- Interpet last 4 digits of the name as MMmm, where MM is the major #; mm is the minor.

## List

[HANDSHAKE\\_V0100](#)

## INIT\_CALLBACK\_STRUCT

Macros for initializing a structure.

## List

[INIT\\_CALLBACK\\_STRUC](#)

## SDKVersion

The SDK version.

## List

[SDK\\_VERSION](#)

## FSDIBEXIFTAG

The exchangeable image file information of camera in JPEG file.

## List



[FS\\_DIB\\_EXIFTAG\\_FLOAT\\_DPIX](#)  
[FS\\_DIB\\_EXIFTAG\\_FLOAT\\_DPIY](#)  
[FS\\_DIB\\_EXIFTAG\\_STRING\\_COPYRIGHT](#)  
[FS\\_DIB\\_EXIFTAG\\_STRING\\_DATETIME](#)  
[FS\\_DIB\\_EXIFTAG\\_STRING\\_IMAGEDESCRIPTION](#)  
[FS\\_DIB\\_EXIFTAG\\_STRING\\_MANUFACTURER](#)  
[FS\\_DIB\\_EXIFTAG\\_STRING\\_MODULE](#)  
[FS\\_DIB\\_EXIFTAG\\_STRING\\_SOFTWARE](#)  
[FS\\_DIB\\_EXIFTAG USHORT\\_ORIENTATION](#)  
[FS\\_DIB\\_EXIFTAG USHORT\\_RESUNIT](#)

## FSDIBConvertFlags

Flags used for converting format.

### List

[FS\\_DIB\\_PALETTE\\_LOC](#)  
[FS\\_DIB\\_PALETTE\\_MAC](#)  
[FS\\_DIB\\_PALETTE\\_WIN](#)

## FSDIBBlendTypes

The flags for bitmap blend.

- - B denotes blend function. It can be Normal, Multiply, Screen...
- - Cr denotes result color components.
- - Cb denotes background color components.
- - Cs denotes source color components.

See [FSDIBitmapCompositeBitmap](#) , [FSDIBitmapCompositeMask](#) .

### List

[FS\\_DIB\\_BLEND\\_COLOR](#)  
[FS\\_DIB\\_BLEND\\_COLORBURN](#)  
[FS\\_DIB\\_BLEND\\_COLORDODGE](#)  
[FS\\_DIB\\_BLEND\\_DARKEN](#)  
[FS\\_DIB\\_BLEND\\_DIFFERENCE](#)  
[FS\\_DIB\\_BLEND\\_EXCLUSION](#)  
[FS\\_DIB\\_BLEND\\_HARDLIGHT](#)  
[FS\\_DIB\\_BLEND\\_HUE](#)  
[FS\\_DIB\\_BLEND\\_LIGHTEN](#)  
[FS\\_DIB\\_BLEND\\_LUMINOSITY](#)  
[FS\\_DIB\\_BLEND\\_MULTIPLY](#)  
[FS\\_DIB\\_BLEND\\_NONSEPARABLE](#)  
[FS\\_DIB\\_BLEND\\_NORMAL](#)  
[FS\\_DIB\\_BLEND\\_OVERLAY](#)  
[FS\\_DIB\\_BLEND\\_SATURATION](#)  
[FS\\_DIB\\_BLEND\\_SCREEN](#)  
[FS\\_DIB\\_BLEND\\_SOFTLIGHT](#)

**FS\_DIB\_BLEND\_UNSUPPORTED****FSDDIBStretchFlags**

Flags used for stretch or transformation.

**List****FS\_DIB\_DOWNSAMPLE**  
**FS\_DIB\_INTERPOL****FSFillingModeFlags**

Flags used for filling page objects.

**List****FSFILL\_ALTERNATE**  
**FSFILL\_FULLSCREEN**  
**FSFILL\_RECT\_AA**  
**FSFILL\_WINDING****FS FileMode**

Definitions for the file mode.

**List****FS\_FILEMODE\_ReadOnly**  
**FS\_FILEMODE\_Truncate**  
**FS\_FILEMODE\_Write****CaptureType**

Definitions for capture type. See [FRCaptureGetType](#) .

**List****FR\_CT\_ANNOTATION**  
**FR\_CT\_TOUCHUP****DocPermission**

Definitions for doc permission. They are the PDF standard permission definitions. You can reference to "PDF Reference - User access permissions".

### List

[FR PERM ANNOTATE](#)  
[FR PERM ASSEMBLE](#)  
[FR PERM EXTRACT ACCESS](#)  
[FR PERM EXTRACT COPY](#)  
[FR PERM FILL FORM](#)  
[FR PERM MODIFY CONTENT](#)  
[FR PERM PRINT](#)  
[FR PERM PRINT HIGH](#)

## FRCIPHERFlag

The flags for cipher.

### List

[FRCIPHER AES](#)  
[FRCIPHER NONE](#)  
[FRCIPHER RC4](#)

## FRDataCollectionActionNames

The definitions of action name for data collection.

### List

[ACTION COPY](#)  
[ACTION CUT](#)  
[ACTION EDIT](#)  
[ACTION FIND](#)  
[ACTION SELECT](#)

## FRDataCollectionFunctionNames

The definitions of function name for data collection.

### List

[FR FUNCTION CALLOUT](#)  
[FR FUNCTION FIND](#)  
[FR FUNCTION HIGHLIGHT](#)

[FR FUNCTION INSERTTEXT](#)  
[FR FUNCTION NOTE](#)  
[FR FUNCTION REPLACETEXT](#)  
[FR FUNCTION SELECTTEXT](#)  
[FR FUNCTION SNAPSHOT](#)  
[FR FUNCTION SQUIGGLYUNDERLINE](#)  
[FR FUNCTION STRIKEOUT](#)  
[FR FUNCTION TEXTBOX](#)  
[FR FUNCTION TYPEWRITER](#)  
[FR FUNCTION UNDERLINE](#)

## FROEMVersion

The definitions for OEM version.

### List

[FR OEM ASUS](#)  
[FR OEM GENERAL](#)  
[FR OEM HPCM](#)  
[FR OEM HPCS](#)  
[FR OEM LENOVO](#)

## FRSIGShowAPFlags

The flag of showing appearance.

### List

[FR SIG SHOW ALL](#)  
[FR SIG SHOW DATE](#)  
[FR SIG SHOW DN](#)  
[FR SIG SHOW FOXITFLOGO](#)  
[FR SIG SHOW LABEL](#)  
[FR SIG SHOW LOCATION](#)  
[FR SIG SHOW NAME](#)  
[FR SIG SHOW REASON](#)

## FRSIGVarifySignatureStates

The state for verifying signature.

### List

[FR SIG TIMESTAMP DOC](#)  
[FR SIG TIMESTAMP EXPIRE](#)  
[FR SIG TIMESTAMP INVALID](#)

[\*\*FR\\_SIG\\_TIMESTAMP\\_ISSUER\\_ISUNKNOWN\*\*](#)  
[\*\*FR\\_SIG\\_TIMESTAMP\\_ISSUER\\_ISINVALID\*\*](#)  
[\*\*FR\\_SIG\\_TIMESTAMP\\_NONE\*\*](#)  
[\*\*FR\\_SIG\\_TIMESTAMP\\_VALID\*\*](#)  
[\*\*FR\\_SIG\\_VERIFY\\_CHANGE\*\*](#)  
[\*\*FR\\_SIG\\_VERIFY\\_ERRORBYTERANGE\*\*](#)  
[\*\*FR\\_SIG\\_VERIFY\\_ERRORDATA\*\*](#)  
[\*\*FR\\_SIG\\_VERIFY\\_INcredible\*\*](#)  
[\*\*FR\\_SIG\\_VERIFY\\_INVALID\*\*](#)  
[\*\*FR\\_SIG\\_VERIFY\\_ISSUER\\_CURRENT\*\*](#)  
[\*\*FR\\_SIG\\_VERIFY\\_ISSUER\\_EXPIRE\*\*](#)  
[\*\*FR\\_SIG\\_VERIFY\\_ISSUER\\_REVOKE\*\*](#)  
[\*\*FR\\_SIG\\_VERIFY\\_ISSUER\\_UNCHECK\*\*](#)  
[\*\*FR\\_SIG\\_VERIFY\\_ISSUER\\_UNKNOWN\*\*](#)  
[\*\*FR\\_SIG\\_VERIFY\\_ISSUER\\_VALID\*\*](#)  
[\*\*FR\\_SIG\\_VERIFY\\_NOSUPPORTWAY\*\*](#)  
[\*\*FR\\_SIG\\_VERIFY\\_VALID\*\*](#)

## SelectionType

Definitions for selection type. See [FRSelectionGetType](#) .

## List

[\*\*FR\\_ST\\_ANNOTATION\*\*](#)  
[\*\*FR\\_ST\\_BITMAP\*\*](#)  
[\*\*FR\\_ST\\_BOOKMARK\*\*](#)  
[\*\*FR\\_ST\\_TEXT\*\*](#)  
[\*\*FR\\_ST\\_THUMBNAIL\*\*](#)

## TaskPaneName

Definitions for task pane name. See [FRAppShowTaskPane](#) .

## List

[\*\*FR\\_TASKPANE\\_ADVSEARCH\*\*](#)

## ActionWizardPresetFlag

## List

[\*\*FR\\_ACTIONWIZARD\\_NOCREATE\*\*](#)  
[\*\*FR\\_ACTIONWIZARD\\_NOPRESET\*\*](#)  
[\*\*FR\\_ACTIONWIZARD\\_PRESET\*\*](#)  
[\*\*FR\\_ACTIONWIZARD\\_PROMPTUSER\*\*](#)

## FRCursorTypeFlags

Flags used for windows stander cursor.

### List

[FR\\_CURSOR\\_TYPE\\_ARROW](#)  
[FR\\_CURSOR\\_TYPE\\_CROSS](#)  
[FR\\_CURSOR\\_TYPE\\_HELP](#)  
[FR\\_CURSOR\\_TYPE\\_SIZEALL](#)  
[FR\\_CURSOR\\_TYPE\\_SIZENESW](#)  
[FR\\_CURSOR\\_TYPE\\_SIZENS](#)  
[FR\\_CURSOR\\_TYPE\\_SIZENWSE](#)  
[FR\\_CURSOR\\_TYPE\\_SIZEWE](#)  
[FR\\_CURSOR\\_TYPE\\_UPARROW](#)  
[FR\\_CURSOR\\_TYPE\\_WAIT](#)

## FRDocTypes

The macro definitions for the types of [FR\\_Document](#) . See [FRDocGetDocumentType](#) .

### List

[FR\\_DOCTYPE\\_DYNAMIC\\_XFA](#)  
[FR\\_DOCTYPE\\_PDF](#)  
[FR\\_DOCTYPE\\_STATIC\\_XFA](#)

## FRMenuEnableNames

The macro definitions for the name of menu you want to enable. See [FRDocSetMenuEnableByName](#) .

### List

[FR\\_MENU\\_ENABLE\\_EMAIL](#)  
[FR\\_MENU\\_ENABLE\\_SAVEAS](#)  
[FR\\_MENU\\_ENABLE\\_SNAPSHOT](#)  
[FR\\_MENU\\_ENABLE\\_STAMP](#)

## FR\_RotationFlags

The rotation flag for document view. See [FRDocViewGetRotation](#) .

### List

[FR\\_ROTATE\\_POS\\_BOTTOM](#)  
[FR\\_ROTATE\\_POS\\_LEFT](#)  
[FR\\_ROTATE\\_POS\\_RIGHT](#)  
[FR\\_ROTATE\\_POS\\_TOP](#)

## EmbedCharSet

### List

[FR\\_EMBED\\_CHARSET](#)

## SpecificChars

### List

[FR\\_VT\\_SECTION\\_FLG](#)  
[FR\\_VT\\_SOFTRETURN](#)  
[FR\\_VT\\_TEXT\\_HIDCHAR](#)  
[FR\\_VT\\_TEXT\\_UNICODEINVALID](#)

## VTWordStyle

### List

[FR\\_VT\\_WORD\\_STYLE\\_BOLD](#)  
[FR\\_VT\\_WORD\\_STYLE\\_ITALIC](#)

## FRFormatToolsButtonIDDefs

The definitions for format tool button ID.

### List

[FR\\_FMT\\_ALIGN\\_CENTER](#)  
[FR\\_FMT\\_ALIGN\\_LEFT](#)  
[FR\\_FMT\\_ALIGN\\_RIGHT](#)  
[FR\\_FMT\\_BOLD](#)  
[FR\\_FMT\\_BORDER\\_COLOR](#)  
[FR\\_FMT\\_CHAR\\_SPACE](#)  
[FR\\_FMT\\_CROSS](#)  
[FR\\_FMT\\_DEDENT](#)  
[FR\\_FMT\\_FILL\\_COLOR](#)  
[FR\\_FMT\\_FONT\\_NAME](#)  
[FR\\_FMT\\_FONT\\_SIZE](#)

[FR\\_FMT\\_HORZ\\_SCALE](#)  
[FR\\_FMT\\_INDENT](#)  
[FR\\_FMT\\_ITALIC](#)  
[FR\\_FMT\\_LINE\\_LEADING](#)  
[FR\\_FMT\\_LINECOLOR](#)  
[FR\\_FMT\\_OPACITY](#)  
[FR\\_FMT\\_SUBSCRIPT](#)  
[FR\\_FMT\\_SUPERSCRIPT](#)  
[FR\\_FMT\\_TEXT\\_COLOR](#)  
[FR\\_FMT\\_UNDERLINE](#)  
[FR\\_FMT\\_WORDSSPACE](#)  
[FR\\_FMT\\_WRITING\\_DIR](#)

## FRMenuNames

The macro definitions for the name of built-in menu in the menu bar. See [FRMenuBarGetMenuItemName](#).

### List

[FR\\_MENU\\_NAME\\_EDIT](#)  
[FR\\_MENU\\_NAME\\_FILE](#)  
[FR\\_MENU\\_NAME\\_HELP](#)  
[FR\\_MENU\\_NAME\\_TOOLS](#)  
[FR\\_MENU\\_NAME\\_VIEW](#)

## FRPanelLocation

### List

[FR\\_PANEL\\_LOCATION\\_LEFT](#)  
[FR\\_PANEL\\_LOCATION\\_TOP](#)

## FRSOURCETYPE

The source type definitions.

### List

[FR\\_SOURCE\\_TYPE\\_ANNOTS](#)  
[FR\\_SOURCE\\_TYPE\\_BOOKMARK](#)  
[FR\\_SOURCE\\_TYPE\\_PAGE](#)  
[FR\\_SOURCE\\_TYPE\\_PAGEOBJECTS](#)  
[FR\\_SOURCE\\_TYPE\\_TAG](#)  
[FR\\_SOURCE\\_TYPE\\_UNKNOWN](#)  
[FR\\_SOURCE\\_TYPE\\_USER](#)

## BuildInToolName

Definitions for tool name.

### List

[FR\\_NAME\\_ADVANCEDSEARCHPAGE](#)  
[FR\\_NAME\\_ANNOT](#)  
[FR\\_NAME\\_AREA](#)  
[FR\\_NAME\\_ARROW](#)  
[FR\\_NAME\\_CALLOUT](#)  
[FR\\_NAME\\_CARET](#)  
[FR\\_NAME\\_CHECKBOX](#)  
[FR\\_NAME\\_CIRCLE](#)  
[FR\\_NAME\\_CLOUDY](#)  
[FR\\_NAME\\_COMBOBOX](#)  
[FR\\_NAME\\_DFA\\_TOOLNAME](#)  
[FR\\_NAME\\_DIMENSION](#)  
[FR\\_NAME\\_DISTANCE](#)  
[FR\\_NAME\\_EDITSELECT](#)  
[FR\\_NAME\\_ELLIPSE](#)  
[FR\\_NAME\\_FAAToolName](#)  
[FR\\_NAME\\_FINDTEXT](#)  
[FR\\_NAME\\_HAND](#)  
[FR\\_NAME\\_HIGHLIGHT](#)  
[FR\\_NAME\\_LINE](#)  
[FR\\_NAME\\_LISTBOX](#)  
[FR\\_NAME\\_LOUPETOOL](#)  
[FR\\_NAME\\_MANGIFIER](#)  
[FR\\_NAME\\_MOVIE](#)  
[FR\\_NAME\\_NOTE](#)  
[FR\\_NAME\\_PENCIL](#)  
[FR\\_NAME\\_PERIMETER](#)  
[FR\\_NAME\\_POLYGON](#)  
[FR\\_NAME\\_POLYLINE](#)  
[FR\\_NAME\\_PUSHBUTTON](#)  
[FR\\_NAME\\_QUADRILATERALLINK](#)  
[FR\\_NAME\\_RADIOBUTTON](#)  
[FR\\_NAME\\_RECTANGLE](#)  
[FR\\_NAME\\_RECTANGLELINK](#)  
[FR\\_NAME\\_REPLACE](#)  
[FR\\_NAME\\_RUBBER](#)  
[FR\\_NAME\\_SCREEN](#)  
[FR\\_NAME\\_SELECTTEXT](#)  
[FR\\_NAME\\_SNAPSHOT](#)  
[FR\\_NAME\\_SOUND](#)  
[FR\\_NAME\\_SQUARE](#)  
[FR\\_NAME\\_SQUIGGLY](#)  
[FR\\_NAME\\_STAMP](#)  
[FR\\_NAME\\_STRIKEOUT](#)  
[FR\\_NAME\\_TEXTBOX](#)  
[FR\\_NAME\\_TEXTFIELD](#)  
[FR\\_NAME\\_TYPEWRITER](#)  
[FR\\_NAME\\_UNDERLINE](#)

## FRToolbarNames

The macro definitions for the name of built-in toolbar. See [FRAppGetToolBarByName](#) .

### List

[FR\\_TOOLBAR\\_NAME\\_ADVANCED\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_BASIC\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_DIGITAL\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_DRAWING2\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_DRAWING\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_FATCH\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_FAVORITE\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_FILE\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_FIND\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_FLYOUTZOOM\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_FORM\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_FORMAT\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_FREETEXT\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_FULLSCREEN\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_LINK\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_MARKUP\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_MARKUPS\\_TOOS](#)  
[FR\\_TOOLBAR\\_NAME\\_MEAS\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_MMEDIA\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_NAVIGATION\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_PROPERTY\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_ROTATEVIEW\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_SECURITY\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_STAMP\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_TEXT\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_TEXTVIEWER\\_TOOLS](#)  
[FR\\_TOOLBAR\\_NAME\\_ZOOM\\_TOOLS](#)

## TextOperations

### List

[FR\\_TEXT\\_OP\\_CREATE](#)  
[FR\\_TEXT\\_OP\\_DELETE](#)  
[FR\\_TEXT\\_OP\\_EDIT](#)  
[FR\\_TEXT\\_OP\\_MERGE](#)  
[FR\\_TEXT\\_OP\\_NONE](#)  
[FR\\_TEXT\\_OP\\_POSTION\\_CHANGE](#)  
[FR\\_TEXT\\_OP\\_RESIZE](#)  
[FR\\_TEXT\\_OP\\_ROTATE](#)  
[FR\\_TEXT\\_OP\\_SHEAR](#)  
[FR\\_TEXT\\_OP\\_SPLIT](#)

## TextOperationUndoRedoType

## List

[FR\\_TEXT\\_OP\\_ON\\_REDO](#)  
[FR\\_TEXT\\_OP\\_ON\\_RELEASE](#)  
[FR\\_TEXT\\_OP\\_ON\\_UNDO](#)

## FPDContentParsingState

The flags for PDF content-parsing state.

## List

[FPD\\_CONTENT\\_NOT\\_PARSED](#)  
[FPD\\_CONTENT\\_PARSED](#)  
[FPD\\_CONTENT\\_PARSING](#)

## FPDOSTextExtractingFlags

The flags for text extracting.

## List

[FPD\\_2TXT\\_AUTO\\_ROTATE](#)  
[FPD\\_2TXT\\_AUTO\\_WIDTH](#)  
[FPD\\_2TXT\\_INCLUDE\\_INVISIBLE](#)  
[FPD\\_2TXT\\_KEEP\\_COLUMN](#)  
[FPD\\_2TXT\\_USE\\_OCR](#)

## FPDTransparencyAttrFlags

Transparency attribute flags for a page object group (form or page).

## List

[FPD\\_TRANS\\_GROUP](#)  
[FPD\\_TRANS\\_ISOLATED](#)  
[FPD\\_TRANS\\_KNOCKOUT](#)

## FRMS\_ACTIVATION\_FLAGS

Activate credential

## List

[FRMS\\_ACTIVATE\\_CREDENTIAL](#)  
[FRMS\\_ACTIVATE\\_MACHINE](#)

## FRMS\_ERR\_CODES

## List

[FRMS\\_NEEDS\\_CREDENTIAL\\_ACTIVATION](#)  
[FRMS\\_NEEDS\\_MACHINE\\_ACTIVATION](#)

## FPDActionTypeNamed

macros define, for action type Named.

## List

[FPD\\_NAMED\\_FIRSTPAGE](#)  
[FPD\\_NAMED\\_LASTPAGE](#)  
[FPD\\_NAMED\\_NEXTPAGE](#)  
[FPD\\_NAMED\\_PREVPAGE](#)

## FPDJavaScriptString

macro for JavaScript string.

## List

[FPD\\_JS\\_MAXLENGTH](#)

## FPDAnnotationFlagBits

Annotation flag bits.

## List

[FPD\\_ANNOTFLAG\\_HIDDEN](#)  
[FPD\\_ANNOTFLAG\\_INVISIBLE](#)  
[FPD\\_ANNOTFLAG\\_LOCKED](#)  
[FPD\\_ANNOTFLAG\\_NORotate](#)  
[FPD\\_ANNOTFLAG\\_NOVIEW](#)  
[FPD\\_ANNOTFLAG\\_NOZOOM](#)  
[FPD\\_ANNOTFLAG\\_PRINT](#)

[\*\*FPD\\_ANNOTFLAG\\_READONLY\*\*](#)  
[\*\*FPD\\_ANNOTFLAG\\_TOGGLENOVIEW\*\*](#)

## FPDBookmarkFontFlags

Bookmark font flags.

### List

[\*\*FPD\\_BOOKMARK\\_BOLD\*\*](#)  
[\*\*FPD\\_BOOKMARK\\_ITALIC\*\*](#)

## FPDConnectedInfoIdTypes

The Id types of connected PDF info object.

### List

[\*\*FPD\\_CONNECT\\_DOCUMENTID\*\*](#)  
[\*\*FPD\\_CONNECT\\_REVIEWID\*\*](#)  
[\*\*FPD\\_CONNECT\\_VERSIONID\*\*](#)

## FPDConnectedInfoTrackingTypes

The tracking types of connected PDF info object.

### List

[\*\*FPD\\_CONNECT\\_BOOKMARKS\*\*](#)  
[\*\*FPD\\_CONNECT\\_COMMENT\*\*](#)  
[\*\*FPD\\_CONNECT\\_COPY\*\*](#)  
[\*\*FPD\\_CONNECT\\_EXTRACT\*\*](#)  
[\*\*FPD\\_CONNECT\\_FORMFILL\*\*](#)  
[\*\*FPD\\_CONNECT\\_PAGES\*\*](#)  
[\*\*FPD\\_CONNECT\\_PRINTING\*\*](#)  
[\*\*FPD\\_CONNECT\\_SIGN\*\*](#)

## FPDConnectedOpenActionURLTypes

The open action URL types of connected PDF info object.

### List

[\*\*FPD\\_CONNECT\\_SUBMIT\\_RECEIVE\*\*](#)

**FPD\_CONNECT\_SUBMIT\_UPDATE****FPDCreatingFlag**

Creating flags for PDF creator.

**List**

[FPDFCREATE\\_INCREMENTAL](#)  
[FPDFCREATE\\_NO\\_ORIGINAL](#)  
[FPDFCREATE\\_OBJECTSTREAM](#)  
[FPDFCREATE\\_PROGRESSIVE](#)

**FPDDestinationZoomMode**

Zoom mode for PDF destination. '/FitB', '/FitBH', '/FitBV' is not supported currently.

**List**

[FPD\\_ZOOM\\_FITHORZ](#)  
[FPD\\_ZOOM\\_FITPAGE](#)  
[FPD\\_ZOOM\\_FITRECT](#)  
[FPD\\_ZOOM\\_FITVERT](#)  
[FPD\\_ZOOM\\_XYZ](#)

**FPDDocPermissions**

PDF document permissions, according to PDF Reference, Table 3.20. 1-2 bits must be zero, 7-8, 13-32 bits must be 1.

**List**

[FPD\\_PERM\\_ANNOT\\_FORM](#)  
[FPD\\_PERM\\_ASSEMBLE](#)  
[FPD\\_PERM\\_EXTRACT](#)  
[FPD\\_PERM\\_EXTRACT\\_ACCESS](#)  
[FPD\\_PERM\\_FILL\\_FORM](#)  
[FPD\\_PERM MODIFY](#)  
[FPD\\_PERM\\_PRINT](#)  
[FPD\\_PERM\\_PRINT\\_HIGH](#)

**FPDDocSaveFlags**

Flags for saving PDF document.

## List

[FPD SAVE DEFAULT](#)  
[FPD SAVE INCREMENTAL](#)  
[FPD SAVE NO ORIGINAL](#)

## FPDLoadErrCodes

The error codes definitions for PDF loading.

## List

[FPD LOADERR\\_ERROR](#)  
[FPD LOADERR\\_FILE](#)  
[FPD LOADERR\\_FORMAT](#)  
[FPD LOADERR\\_MEMORY](#)  
[FPD LOADERR\\_NOTFOUND](#)  
[FPD LOADERR\\_PARAM](#)  
[FPD LOADERR\\_PASSWORD](#)  
[FPD LOADERR\\_STATUS](#)  
[FPD LOADERR\\_SUCCESS](#)  
[FPD LOADERR\\_TOBECONTINUED](#)

## FPDFontColorSpaceFamilies

Color space families. See [FPDCOLORSPACEGETFAMILY](#).

## List

[FPD\\_CS\\_CALGRAY](#)  
[FPD\\_CS\\_CALRGB](#)  
[FPD\\_CS\\_DEVICECMYK](#)  
[FPD\\_CS\\_DEVICEGRAY](#)  
[FPD\\_CS\\_DEVICEN](#)  
[FPD\\_CS\\_DEVICERGB](#)  
[FPD\\_CS\\_ICCBASED](#)  
[FPD\\_CS\\_INDEXED](#)  
[FPD\\_CS\\_LAB](#)  
[FPD\\_CS\\_PATTERN](#)  
[FPD\\_CS\\_SEPARATION](#)

## FPDFontFlags

Font flags. See [FPDFONTGETFLAGS](#).

## List

[FPD FONT ALLCAP](#)  
[FPD FONT FIXEDPITCH](#)  
[FPD FONT FORCEBOLD](#)  
[FPD FONT ITALIC](#)  
[FPD FONT NONSYMBOLIC](#)  
[FPD FONT SCRIPT](#)  
[FPD FONT SERIF](#)  
[FPD FONT SMALLCAP](#)  
[FPD FONT SYMBOLIC](#)

## FPDFontPredefinedEncoding

Predefined encoding. See [FPDFontEncodingNew](#).

### List

[FPD FONT ENCODING ADOBE SYMBOL](#)  
[FPD FONT ENCODING BUILTIN](#)  
[FPD FONT ENCODING MACEXPERT](#)  
[FPD FONT ENCODING MACROMAN](#)  
[FPD FONT ENCODING MS SYMBOL](#)  
[FPD FONT ENCODING PDFDOC](#)  
[FPD FONT ENCODING STANDARD](#)  
[FPD FONT ENCODING UNICODE](#)  
[FPD FONT ENCODING WINANSI](#)  
[FPD FONT ENCODING ZAPFDINGBATS](#)

## FPDFontTypeIDs

Font type IDs. See [FPDFontGetType](#).

### List

[FPD FONT CIDFONT](#)  
[FPD FONT TRUEETYPE](#)  
[FPD FONT TYPE1](#)  
[FPD FONT TYPE3](#)

## FPDFieldTypes

macros definition for field types in CreateControl method. Note: They are used in some interfaces about FPD\_InterForm, especially for JavaScript.

### List

[FPD FORM FIELDTYPE CHECKBOX](#)  
[FPD FORM FIELDTYPE COMBOBOX](#)

[\*\*FPD FORM FIELDTYPE LISTBOX\*\*](#)  
[\*\*FPD FORM FIELDTYPE PUSHBUTTON\*\*](#)  
[\*\*FPD FORM FIELDTYPE RADIobutton\*\*](#)  
[\*\*FPD FORM FIELDTYPE SIGNATURE\*\*](#)  
[\*\*FPD FORM FIELDTYPE TEXTFIELD\*\*](#)  
[\*\*FPD FORM FIELDTYPE UNKNOWN\*\*](#)

## FPDComboBoxFieldsAdditionalFlags

Additional flags for combo box fields.

### List

[\*\*FPD FORM COMBO EDIT\*\*](#)

## FPDFormFieldFlags

Form field flags.

### List

[\*\*FPD FORM FIELD NOEXPORT\*\*](#)  
[\*\*FPD FORM FIELD READONLY\*\*](#)  
[\*\*FPD FORM FIELD REQUIRED\*\*](#)

## FPDFormFieldTextPosition

Text Position definition, for "TP" entry in MK of Widget dictionary.

### List

[\*\*FPD TEXT POS ABOVE\*\*](#)  
[\*\*FPD TEXT POS BELOW\*\*](#)  
[\*\*FPD TEXT POS CAPTION\*\*](#)  
[\*\*FPD TEXT POS ICON\*\*](#)  
[\*\*FPD TEXT POS LEFT\*\*](#)  
[\*\*FPD TEXT POS OVERLAID\*\*](#)  
[\*\*FPD TEXT POS RIGHT\*\*](#)

## FPDListBoxFieldsAdditionalFlags

Additional flags for list box fields.

### List

**FPD FORM LIST MULTISELECT****FPDRadioButtonAdditionalFlags**

Additional flags for radio button fields.

**List**

**FPD FORM RADIO NOTOGGLEOFF**  
**FPD FORM RADIO UNISON**

**FPDTextFieldsAdditionalFlags**

Additional flags for text fields.

**List**

**FPD FORM TEXT COMB**  
**FPD FORM TEXT MULTILINE**  
**FPD FORM TEXT NOSCROLL**  
**FPD FORM TEXT PASSWORD**

**FPDFXEncodingType**

Encoding type. See [FPDFXFontEncodingGlyphFromCharCodeEx](#) .

**List**

**FPD FXENCODING INTERNAL**  
**FPD FXENCODING UNICODE**

**FPDInterFormColorTypes**

macros definition for color types in interactive form. used for color operator like G/g, RG/rg, K/k, or "BC"/"BG" entries in MK of Widget dictionary, etc...

**List**

**FPD CLRTYPE CMYK**  
**FPD CLRTYPE GRAY**  
**FPD CLRTYPE RGB**  
**FPD CLRTYPE TRANSPARENT**

**FPDObjTypes**

Object types in PDF. Return by FPDOBJECTGETTYPE() function.

## List

[FPD\\_OBJ\\_ARRAY](#)  
[FPD\\_OBJ\\_BOOLEAN](#)  
[FPD\\_OBJ\\_DICTIONARY](#)  
[FPD\\_OBJ\\_INVALID](#)  
[FPD\\_OBJ\\_NAME](#)  
[FPD\\_OBJ\\_NULL](#)  
[FPD\\_OBJ\\_NUMBER](#)  
[FPD\\_OBJ\\_REFERENCE](#)  
[FPD\\_OBJ\\_STREAM](#)  
[FPD\\_OBJ\\_STRING](#)

## FPDPageObjectType

The types of page objects.

## List

[FPD\\_PAGEOBJ\\_FORM](#)  
[FPD\\_PAGEOBJ\\_IMAGE](#)  
[FPD\\_PAGEOBJ\\_INLINES](#)  
[FPD\\_PAGEOBJ\\_PATH](#)  
[FPD\\_PAGEOBJ\\_SHADING](#)  
[FPD\\_PAGEOBJ\\_TEXT](#)

## FPDPathPointFlags

Point flags used in FPD\_Path.

## List

[FPDPT\\_BEZIERTO](#)  
[FPDPT\\_CLOSEFIGURE](#)  
[FPDPT\\_LINETO](#)  
[FPDPT\\_MOVETO](#)  
[FPDPT\\_TYPE](#)

## FPDParseContextFlags

Flags for parser context.

## List

[\*\*FPD PARSE NOSTREAM\*\*](#)  
[\*\*FPD PARSE TYPEONLY\*\*](#)

## FPDParseErrCodes

Reasons for PDF parsing failure: returned by [FPDParserStartParse \(\)](#) function..

### List

[\*\*FPD PARSE ERROR CERT\*\*](#)  
[\*\*FPD PARSE ERROR FILE\*\*](#)  
[\*\*FPD PARSE ERROR FORMAT\*\*](#)  
[\*\*FPD PARSE ERROR HANDLER\*\*](#)  
[\*\*FPD PARSE ERROR PASSWORD\*\*](#)  
[\*\*FPD PARSE ERROR SUCCESS\*\*](#)

## FPDSearchFlags

Search flags. See [FPDTextPageFindFindFirst](#) , [FPDTextPageFindFindNext](#) ,  
[FPDTextPageFindFindPrev](#) .

### List

[\*\*FPD TEXT MATCHCASE\*\*](#)  
[\*\*FPDTEXT CONSECUTIVE\*\*](#)  
[\*\*FPDTEXT MATCHWHOLEWORD\*\*](#)

## FPDSearchingStatus

Searching status for progressive searching. See [FPDProgressiveSearchGetStatus](#) .

### List

[\*\*FPD SCH STATUS FAILED\*\*](#)  
[\*\*FPD SCH STATUS FOUND\*\*](#)  
[\*\*FPD SCH STATUS NOTFOUND\*\*](#)  
[\*\*FPD SCH STATUS READY\*\*](#)  
[\*\*FPD SCH STATUS TOBECONTINUED\*\*](#)

## FPDRenderOptBitmasksFlag

Flag bitmasks for render option. See [FPDRenderOptionsGetRenderFlag](#) ,  
[FPDRenderOptionsSetRenderFlag](#) .

## List

[\*\*FPD\\_RENDER\\_BGR\\_STRIPE\*\*](#)  
[\*\*FPD\\_RENDER\\_CLEARTYPE\*\*](#)  
[\*\*FPD\\_RENDER\\_FORCE\\_DOWNSAMPLE\*\*](#)  
[\*\*FPD\\_RENDER\\_FORCE\\_HALFTONE\*\*](#)  
[\*\*FPD\\_RENDER\\_LIMITEDIMAGECACHE\*\*](#)  
[\*\*FPD\\_RENDER\\_PRINTGRAPHICTEXT\*\*](#)  
[\*\*FPD\\_RENDER\\_PRINTPREVIEW\*\*](#)  
[\*\*FPD\\_RENDER\\_RECT\\_AA\*\*](#)

## FPDRenderOptColorModeCodes

Color mode codes for render options. See [FPDRenderOptionsGetColorMode](#) , [FPDRenderOptionsSetColorMode](#) .

## List

[\*\*FPD\\_RENDER\\_COLOR\\_ALPHA\*\*](#)  
[\*\*FPD\\_RENDER\\_COLOR\\_GRAY\*\*](#)  
[\*\*FPD\\_RENDER\\_COLOR\\_NORMAL\*\*](#)  
[\*\*FPD\\_RENDER\\_COLOR\\_TWOCOLOR\*\*](#)

## FPDCharInfoFlag

char info flag.

## List

[\*\*FPD\\_CHAR\\_ERROR\*\*](#)  
[\*\*FPD\\_CHAR\\_GENERATED\*\*](#)  
[\*\*FPD\\_CHAR\\_NORMAL\*\*](#)  
[\*\*FPD\\_CHAR\\_UNUNICODE\*\*](#)

## FPDTextDirectionsType

Directions type. See [FPDTextGetOrderByDirection](#) .

## List

[\*\*FPD\\_TEXT\\_DOWN\*\*](#)  
[\*\*FPD\\_TEXT\\_LEFT\*\*](#)  
[\*\*FPD\\_TEXT\\_RIGHT\*\*](#)  
[\*\*FPD\\_TEXT\\_UP\*\*](#)

## FPDTextPageFlags

Flags for creating text page. See [FPDTextNew](#) .

## List

[\*\*FPD\\_TEXT\\_DISPLAY\\_ORDER\*\*](#)  
[\*\*FPD\\_TEXT\\_STREAM\\_ORDER\*\*](#)

## FPDWrapperDocGetWrapperType

The Id types of connected PDF info object.

## List

[\*\*FPD\\_WRAPPERTYPE\\_FOXIT\*\*](#)  
[\*\*FPD\\_WRAPPERTYPE\\_NO\*\*](#)  
[\*\*FPD\\_WRAPPERTYPE\\_PDF2\*\*](#)

# Callback Groups

---

## PIHandshakeData\_V0100

PIHandshakeData\_V0100 Structure containing data to be passed in and out of the plug-ins PIHandshake() routine. This routine MUST be implemented with that name by the plug-in writer. The first field of any PIHandshakeData\_VMMmm struct must be handshakeVersion.

## Syntax

```
typedef struct {
    unsigned long lStructSize;
    PIHDPltfmData *pPltfmData;
    FS_INT32 (*PIHDGetHandshakeVersion)(
        );
    char* (*PIHDGetAppName)(
        );
    FS_BOOL (*PIHDRegisterPlugin)(
        void* thisData,
        char* name,
        FS_LPCWSTR lpwsTitle
        );
    void (*PIHDSetExportHFTsCallback)(
        void* thisData,
        PIExportHFTsProcType proc
    );
}
```

```

);
void (*PIHDSetImportReplaceAndRegisterCallback)(
    void* thisData,
    PImportReplaceAndRegisterProcType proc
);

void (*PIHDSetInitDataCallback)(
    void* thisData,
    PIInitDataProcType proc
);

void (*PIHDSetInitUICallbacks)(
    void* thisData,
    PIInitUIProcs* pProcs
);

void (*PIHDSetUnloadCallback)(
    void* thisData,
    PIUnloadProcType proc
);

} PIHandshakeData_V0100;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( PIHandshakeData\_V0100 )*.

**\*pPltfmData**

Data for plug-in platform

**PIInitUIProcs**

Structure containing callbacks to initialize the UI. Invokes [INIT\\_CALLBACK\\_STRUCT](#) to initialize the callbacks.

**Syntax**

```

typedef struct {
    unsigned long IStructSize;
    void (*PILoadMenuBarUI)(
        void* pParentWnd
    );

    void (*PIReleaseMenuBarUI)(
        void* pParentWnd
    );

    void (*PILoadToolBarUI)(
        void* pParentWnd
    );

```

```

void (*PIReleaseToolBarUI)(
    void* pParentWnd
);

void (*PILoadRibbonUI)(
    void* pParentWnd
);

void (*PILoadStatusBarUI)(
    void* pParentWnd
);

void (*PIReleaseStatusBarUI)(
    void* pParentWnd
);

void (*PIReleaseRibbonUI)(
    void* pParentWnd
);

} PIInitUIProcs;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( PIInitUIProcs )*.

**PISDKData\_V0100**

Structure containing callbacks to get or send data to the plug-in's PISetupSDK() routine. This routine is implemented in the PIMain.c routine supplied with the Plug-in SDK. The first field of any SDKData\_VMMmm struct must be handshakeVersion.

**Syntax**

```

typedef struct {
    unsigned long IStructSize;
    FS_INT32 (*PISDGetHandshakeVersion)(
        );
    void (*PISDSetHandshakeProc)(
        void* thisData,
        PIHandshakeProcType proc
        );
    FRCOREHFTMGR* (*PISDGetCoreHFT)(
        );
    FS_LPVVOID (*PISDGetPID)(
        void* thisData
        );
    void (*PISetSDKVersion)(
        void* thisData,
        
```

```

FS_INT32 nPISDKVersion
);

void (*PISDSetMakeTokenProc)(
    void* thisData,
    PIMakeTokenProcType proc
);

} PISDKData_V0100;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( [PISDKData\\_V0100](#) )*.

**PIHDPltfmData**

PIHDPltfmData Structure containing data to exchange with plug-in platform and will not be passed to common plug-in.

**Syntax**

```

typedef struct {
    FS\_BOOL (*PIHDLoadPlugin)(
        void* arrLibPath,
        FS\_BOOL bFreeLibWhenFailed
    );

    void (*PIHDSetOnAboutPluginsCallback)(
        PIOOnAboutPluginsProcType proc
    );

    FS\_INT32 (*PIHDGetLastPluginLoadError)(
    );

    FS\_BOOL (*PIHDIsPluginDisabledBy)(
        FS\_LPCWSTR lpwsFileName
    );

    void (*PIHDSetOnDelayLoadJSPluginsCallback)(
        PIOOnDelayLoadJSPluginsProcType proc
    );

    FS\_BOOL (*PIHDLoadPluginUI)(
        HWND hWnd,
        void* arrLibPath
    );
}

} PIHDPltfmData;

```

**FS\_FileRead**

## Syntax

```
typedef struct {
    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_BOOL (*FSFileRead_ReadBlock)(FS_LPVOID clientData,
                                         void* buffer,
                                         FS_DWORD offset,
                                         FS_DWORD size
                                         );
    FS_DWORD (*FSFileRead_GetSize)(FS_LPVOID clientData
                                         );
    void (*FSFileRead_Release)(FS_LPVOID clientData
                                         );
} FS_FileRead;
```

### IStructSize

The size of data structure. It must be set to *sizeof*( [FS\\_FileRead](#) ).

### clientData

The user-supplied data.

## FS\_FileStreamCallbacksRec

A callback set for file stream reading and writing.

## Syntax

```
typedef struct _fs_filestream_callbacks_ {
    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_FileStream (*FSFileStreamRetain)(FS_LPVOID clientData
                                         );
    void (*FSFileStreamRelease)(FS_LPVOID clientData
                                         );
    FS_INT32 (*FSFileStreamGetSize)(FS_LPVOID clientData
                                         );
    FS_BOOL (*FSFileStreamIsEOF)(FS_LPVOID clientData
                                         );
```

```

);
FS\_INT32 (*FSFileStreamGetPosition)(
    FS\_LPVOID clientData
);
FS\_BOOL (*FSFileStreamReadBlock)(
    FS\_LPVOID clientData,
    void* buffer,
    FS\_INT32 offset,
    unsigned int size
);
unsigned int (*FSFileStreamReadBlock2)(
    FS\_LPVOID clientData,
    void* buffer,
    unsigned int size
);
FS\_BOOL (*FSFileStreamWriteBlock)(
    FS\_LPVOID clientData,
    const void* buffer,
    FS\_INT32 offset,
    unsigned int size
);
FS\_BOOL (*FSFileStreamFlush)(
    FS\_LPVOID clientData
);
}

} FS_FileStreamCallbacksRec, *FS_FileStreamCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FS\_FileStreamCallbacksRec )*.

**clientData**

The user-supplied data.

**FS\_FileWrite****Syntax**

```

typedef struct {
    unsigned long IStructSize;
    FS\_LPVOID clientData;
    FS\_DWORD (*FSFileWrite\_GetSize)(
        FS\_LPVOID clientData
    );
    FS\_DWORD (*FSFileWrite\_Flush)(
        FS\_LPVOID clientData
    );
}

```

```

FS_BOOL (*FSFileWrite_WriteBlock)(  

    FS_LPVOID clientData,  

    const void* pData,  

    FS_DWORD offset,  

    FS_DWORD size  

);  
  

void (*FSFileWrite_Release)(  

    FS_LPVOID clientData  

);  
  

} FS_FileWrite;
```

**IStructSize**

The size of data structure. It must be set to *sizeof( FS\_FileWrite )*.

**clientData**

The user-supplied data.

**FS\_Pause**

Simple pause interface.

**Syntax**

```

typedef struct {  

    unsigned long IStructSize;  

    FS_BOOL (*NeedToPauseNow)(  

        void  

    );  
  

} FS_Pause, *PFS_Pause;
```

**IStructSize**

The size of data structure. It must be set to *sizeof( FS\_Pause )*.

**FS\_StreamWrite****Syntax**

```

typedef struct {  

    unsigned long IStructSize;  

    FS_LPVOID clientData;  

    FS_BOOL (*FSSStreamWrite_WriteBlock)(  

        FS_LPVOID clientData,
```

```

        const void* pData,
FS_DWORD size
);

void (*FSStreamWrite\_Release)(
    FS_LPVOID clientData
);
}

} FS_StreamWrite;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FS\_StreamWrite )*.

**clientData**

The user-supplied data.

**FR\_ActionHandlerCallbacksRec**

The callbacks for action handler. The callbacks are called by Foxit Reader when the Foxit Reader will do the actions. You can implement your own process to customize the action process.

**Syntax**

```

typedef struct _fr_actionhandler_callbacks_{

    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_BOOL (*FRActionHandlerDocOpen)(

        FS_LPVOID clientData,
        FPD_Action action,
        FR_Document frDoc,
        FR_DocView frDocView,
        FS_BOOL bDisableGoto
    );

    FS_BOOL (*FRActionHandlerJavaScript)(

        FS_LPVOID clientData,
        FPD_Action action,
        FS_LPCWSTR lpwsJSName,
        FR_Document frDoc,
        FR_DocView frDocView
    );

    FS_BOOL (*FRActionHandlerPage)(

        FS_LPVOID clientData,
        FPD_Action action,
        FPD_AActionType type,
        FR_Document frDoc,
        FR_DocView frDocView
    );

    FS_BOOL (*FRActionHandlerLink)(


```

```

    FS_LPVOID clientData,
    FPD_Action action,
    FR_Document frDoc,
    FR_DocView frDocView
);

} FR_ActionHandlerCallbacksRec, *FR_ActionHandlerCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof*( [FR\\_ActionHandlerCallbacksRec](#) ).

**clientData**

The user-supplied data.

## FR\_AppEventCallbacksRec

A callback set for application-level event handler. See [FRAppRegisterAppEventHandler](#) .

### Syntax

```

typedef struct _fr_appevent_callbacks_{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    void (*FRAppOnLangUIChange)(
        FS_LPVOID clientData
    );

    void (*FRAppOnActivate)(
        FS_LPVOID clientData,
        FS_BOOL bActive
    );

    void (*FRAppWillQuit)(
        FS_LPVOID clientData
    );

    void (*FRAppWillQuit)(
        FS_LPVOID clientData,
        FS_LPCSTR lpModuleName
    );

    void (*FRAppOnToolChanged)(
        FS_LPVOID clientData,
        FR_Tool deactivateTool,
        FR_Tool activeTool
    );

    void (*FRAppOnSkinChange)(
        FS_LPVOID clientData
    );

    void (*FRAppOnShowFullScreen)(

```

```
FS_LPVOID clientData
);

void (*FRAppOnHideFullScreen)(
    FS_LPVOID clientData
);

void (*FRAppOnResetToolbars)(
    FS_LPVOID clientData
);

void (*FRAppOnRunCommandLine)(
    FS_LPVOID clientData
);

void (*FRAppOnDrop)(
    FS_LPVOID clientData,
    FS_LPCWSTR lpszPath
);

void (*FRAppOnTextViewChange)(
    FS_LPVOID clientData,
    FS_INT32 nViewType
);

void (*FRAppOnRecentFileListChange)(
    FS_LPVOID clientData,
    FS_WideStringArray arrFileList
);

void (*FRAppOnShowRibbonFilePage)(
    FS_LPVOID clientData
);

void (*FRAppOnCloseRibbonFilePage)(
    FS_LPVOID clientData
);

void (*FRAppOnRibbonElementsLoadFinish)(
    FS_LPVOID clientData,
    void* pParentWnd
);

FS_BOOL (*FRAppOnOpenDocument)(
    FS_LPVOID clientData,
    FS_LPCWSTR lpszFilePath
);

void (*FRAppOnRibbonCategoryClicked)(
    FS_LPVOID clientData,
    FS_BOOL bChanged
);

void (*FRAppOnMainFrmLoadFinish)(
    FS_LPVOID clientData
);

void (*FRAppOnRibbonUILayoutFinish)(
```

---

```

    FS_LPVOID clientData,
    void* pParentWnd
);

void (*FRAppOnShowTaskPane)(  

    FS_LPVOID clientData,  

    FS_BOOL bShow  

);

void (*FRAppOnShowPopupMenu)(  

    FS_LPVOID clientData  

);

void (*FRAppOnDidOpenAllFiles)(  

    FS_LPVOID clientData  

);

void (*FRAppOnMainFrameWillClose)(  

    FS_LPVOID clientData,  

    FS_BOOL* bCanClose  

);

void (*FRAppMainFrameOnSize)(  

    FS_LPVOID clientData  

);

void (*FRAppMainFrameOnLoadWinPlacementFinish)(  

    FS_LPVOID clientData,  

    HWND hMainframe  

);

void (*FRAppPluginOnLoaded)(  

    FS_LPVOID clientData,  

    FS_LPCWSTR lpszPluginName  

);

void (*FRAppOnDidCloseRibbonFilePage)(  

    FS_LPVOID clientData  

);

} FR_AppEventCallbacksRec, *FR_AppEventCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof* ([FR\\_AppEventCallbacksRec](#)).

**clientData**

The user-supplied data.

## FR\_CaptureCallbacksRec

A callback set for capture handler. These callbacks implement a capture handler. The callbacks implement the capture handler functions. See [FRAppRegisterCaptureHandler](#).

## Syntax

```
typedef struct _fr_capture_callbacks{
    unsigned long lStructSize;
    FS_LPVOID clientData;
    FS_LPSTR (*FRCaptureGetType)(
        FS_LPVOID clientData
    );

    void (*FRCaptureLosingCapture)(
        FS_LPVOID clientData,
        FR_Document document,
        void* curCaptureData
    );

    void (*FRCaptureGettingCapture)(
        FS_LPVOID clientData,
        FR_Document document,
        void* curCaptureData
    );

    FS_BOOL (*FRCaptureLButtonDown)(
        FS_LPVOID clientData,
        FR_PageView pageView,
        void* curCaptureData,
        unsigned int nFlags,
        FS_DevicePoint point
    );

    FS_BOOL (*FRCaptureLButtonUp)((
        FS_LPVOID clientData,
        FR_PageView pageView,
        void* curCaptureData,
        unsigned int nFlags,
        FS_DevicePoint point
    );

    FS_BOOL (*FRCaptureLButtonDblClk)((
        FS_LPVOID clientData,
        FR_PageView pageView,
        void* curCaptureData,
        unsigned int nFlags,
        FS_DevicePoint point
    );

    FS_BOOL (*FRCaptureMouseMove)((
        FS_LPVOID clientData,
        FR_PageView pageView,
        void* curCaptureData,
        unsigned int nFlags,
        FS_DevicePoint point
    );

    FS_BOOL (*FRCaptureRButtonDown)((
        FS_LPVOID clientData,
        FR_PageView pageView,
        void* curCaptureData,
        unsigned int nFlags,
```

---

```

FS_DevicePoint point
);

FS_BOOL (*FRCaptureRButtonUp)(
    FS_LPVOID clientData,
    FR_PageView pageView,
    void* curCaptureData,
    unsigned int nFlags,
    FS_DevicePoint point
);

FS_BOOL (*FRCaptureRButtonDblClk)(
    FS_LPVOID clientData,
    FR_PageView pageView,
    void* curCaptureData,
    unsigned int nFlags,
    FS_DevicePoint point
);

FS_BOOL (*FRCaptureContextMenu)(
    FS_LPVOID clientData,
    FR_PageView pageView,
    FS_DevicePoint point
);

} FR_CaptureCallbacksRec, *FR_CaptureCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof* ([FR\\_CaptureCallbacksRec](#)).

**clientData**

The user-supplied data.

**FR\_ContentProviderCallbacksRec**

The structure containing content provider callbacks. Plug-ins invoke these callbacks to provide PDF content to Foxit Reader. These PDF files are usually not in standard structure and may be encrypted by customer algorithm. See

[FRApplRegisterContentProvider](#).

**Syntax**

```

typedef struct _fr_content_provider_callbacks_{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_BOOL (*FRConProviderOnFileOpen)(
        FS_LPVOID clientData,
        FR_Document doc,
        FS_LPCWSTR lpszSource,
        FS_BOOL bIsAttachment
    );
}

```

```
unsigned long (*FRConProviderOnGetPermissions)(  
    FS_LPVOID clientData,  
    FR_Document doc,  
    unsigned long pdfselfPermissions  
);  
  
FS_BOOL (*FRConProviderOnGetContentSize)(  
    FS_LPVOID clientData,  
    FR_Document doc,  
    unsigned long* pTotalSize  
);  
  
FS_BOOL (*FRConProviderOnReadContent)(  
    FS_LPVOID clientData,  
    FR_Document doc,  
    FS_DWORD pos,  
    unsigned char* pBuf,  
    unsigned long size  
);  
  
FS_BOOL (*FRConProviderOnWriteContent)(  
    FS_LPVOID clientData,  
    FR_Document doc,  
    unsigned char* pBuf,  
    unsigned long size,  
    FS_LPCWSTR lpSaveFilePath  
);  
  
void (*FRConProviderOnFileClose)(  
    FS_LPVOID clientData,  
    FR_Document doc  
);  
  
FS_BOOL (*FRConProviderOnWriteAttachmentContent)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    unsigned char * pBuf,  
    unsigned long size,  
    FS_LPCWSTR lpAttachmntPath  
);  
  
FS_BOOL (*FRConProviderOnGetAttachmentSize)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    unsigned long* pTotalSize,  
    FS_LPCWSTR lpAttachmntPath  
);  
  
FS_BOOL (*FRConProviderOnReadAttachmentContent)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    unsigned char* pBuf,  
    unsigned long size,  
    FS_LPCWSTR lpAttachmntPath  
);  
  
FS_BOOL (*FRConProviderIsPageAvail)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,
```

```
FS_INT32 iPage,
clientData,
frDoc,
iPage,
FS_FLOAT* outWidth,
FS_FLOAT* outHeight,
clientData,
frDoc,
iPage,
FS_DIBitmap bitmap,
FS_INT32 start_x,
FS_INT32 start_y,
FS_INT32 width,
FS_INT32 height
);

FS_BOOL (*FRConProviderNeedReopenDoc)(
    FS_LPVOID clientData,
    FR_Document frDoc
);

FS_BOOL (*FRConProviderCanBeSaved)(
    FS_LPVOID clientData,
    FR_Document frDoc
);

FS_BOOL (*FRContentProviderIsProcessErrMsg)(
    FS_LPVOID clientData
);

FS_BOOL (*FRContentProviderIsSupportPanel)(
    FS_LPVOID clientData,
    FR_Document frDoc
);

FS_BOOL (*FRContentProviderIsSupportViewByScroll)(
    FS_LPVOID clientData,
    FR_Document frDoc
);

FS_BOOL (*FRContentProviderIsSupportViewByScroll)(
    FS_LPVOID clientData,
    FS_LPCSTR lpsName
);

FS_BOOL (*FROnGetPageDictObjectNumber)(
    FS_LPVOID clientData,
    FR_Document frDoc,
    FS_INT32 nPageIndex,
    FS_DWORD* outObjNumber
);

FS_BOOL (*FRConProviderIsWriteContentProgressive)(
    FS_LPVOID clientData,
    FR_Document doc,
    FS_LPCWSTR lpSaveFilePath
);
```

---

```

FS_BOOL (*FRCOnProviderWriteContentProgressiveFinish)(  

    FS_LPVOID clientData,  

    FR_Document doc,  

    FS_LPCWSTR lpSaveFilePath,  

    FS_BOOL bResult  

);

FS_BOOL (*FRCOnProviderIsWriteAttachmentContent)(  

    FS_LPVOID clientData,  

    FR_Document frDoc,  

    FS_LPCWSTR lpAttachmntPath  

);

FS_BOOL (*FRCOnProviderOnWriteAttachmentContentFinish)(  

    FS_LPVOID clientData,  

    FR_Document frDoc,  

    FS_LPCWSTR lpAttachmntPath  

);

FS_BOOL (*FRCOnProviderOnReadAttachmentContentInBlocks)(  

    FS_LPVOID clientData,  

    FR_Document frDoc,  

    FS_DWORD pos,  

    unsigned char* pBuf,  

    unsigned long size,  

    FS_LPCWSTR lpAttachmntPath  

);

FS_BOOL (*FRCOnProviderOnBackFillContent)(  

    FS_LPVOID clientData,  

    FR_Document doc,  

    unsigned char* pBuf,  

    unsigned long size,  

    FS_LPCWSTR lpSaveFilePath  

);

} FR_ContentProviderCallbacksRec,  

*FR_ContentProviderCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof*( [FR\\_ContentProviderCallbacksRec](#) ).

**clientData**

The user-supplied data.

## FR\_CryptoCallbacksRec

The structure containing the crypto handler callback functions. They are for PDF cryptographic operations (encryption and decryption). The crypto handler works with security handler which provides algorithm and key info.

### Syntax

```

typedef struct _fr_crypto_callbacks_ {
    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_LPVOID (*FREncryptCreateHandler)(FS_LPVOID clientData,
                                           FPD_Object encryptDict,
                                           FS_LPVOID securityData
                                         );
    FS_DWORD (*FREncryptDecryptGetSize)(FS_LPVOID clientData,
                                           FS_LPVOID cryptoHandler,
                                           FS_DWORD src_size
                                         );
    FS_LPVOID (*FREncryptDecryptStart)(FS_LPVOID clientData,
                                           FS_LPVOID cryptoHandler,
                                           FS_DWORD objnum,
                                           FS_DWORD genum
                                         );
    FS_BOOL (*FREncryptDecryptStream)(FS_LPVOID clientData,
                                         FS_LPVOID cryptoContext,
                                         FS_LPCBYTE src_buf,
                                         FS_DWORD src_size,
                                         FS_BinaryBuf dest_buf
                                         );
    FS_BOOL (*FREncryptDecryptFinish)(FS_LPVOID clientData,
                                         FS_LPVOID cryptoContext,
                                         FS_BinaryBuf dest_buf
                                         );
    FS_DWORD (*FREncryptEncryptGetSize)(FS_LPVOID clientData,
                                           FS_LPVOID cryptoHandler,
                                           FS_DWORD objnum,
                                           FS_DWORD version,
                                           FS_LPCBYTE src_buf,
                                           FS_DWORD src_size
                                         );
    FS_BOOL (*FREncryptEncryptContent)(FS_LPVOID clientData,
                                          FS_LPVOID cryptoHandler,
                                          FS_INT32 objnum,
                                          unsigned long version,
                                          FS_LPCBYTE src_buf,
                                          FS_DWORD src_size,
                                          FS_LPBYTE dest_buf,
                                          FS_DWORD* outDestsize
                                         );
    void (*FREncryptFinishHandler)(FS_LPVOID clientData,
                                   FS_LPVOID cryptoHandler
                                 );
}

```

```

    );
FS\_BOOL (*FREncryptProgressiveEncryptStart)(
    FS\_LPVOID clientData,
    FS\_LPVOID cryptoHandler,
    FS\_DWORD objnum,
    unsigned long version,
    FS\_DWORD raw_size,
    FS\_BOOL bFlateEncode
);

FS\_BOOL (*FREncryptProgressiveEncryptContent)(
    FS\_LPVOID clientData,
    FS\_LPVOID cryptoHandler,
    FS\_INT32 objnum,
    unsigned long version,
    FS\_LPCBYTE src_buf,
    FS\_DWORD src_size,
    FS\_BinaryBuf dest_buf
);

FS\_BOOL (*FREncryptProgressiveEncryptFinish)(
    FS\_LPVOID clientData,
    FS\_LPVOID cryptoHandler,
    FS\_BinaryBuf dest_buf
);

} FR_CryptoCallbacksRec, *FR_CryptoCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FR\_CryptoCallbacksRec )*.

**clientData**

The user-supplied data.

**FR\_DataCollectionHandlerCallbacksRec**

The callbacks for data collection handler. The callbacks are called by Foxit Reader when some action occurs. Then you can implement the callbacks to collect the data.

**Syntax**

```

typedef struct _fr_datacollectionhandler_callbacks_{
    unsigned long IStructSize;
    FS\_LPVOID clientData;
    void (*FROnCollectNormalData)(
        FS\_LPVOID clientData,
        FS\_LPCWSTR lpwsFunction,
        FS\_LPCWSTR lpwsAction,
        FS\_LPCWSTR lpwsContent
    );

```

```

void (*FROnCollectBitmapData)(  

    FS\_LPVOID clientData,  

    FS\_LPCWSTR lpwsFunction,  

    FS\_LPCWSTR lpwsAction,  

    FS\_DIBitmap bitmap  

);  
  

void (*FROnCollectNormalData2)(  

    FS\_LPVOID clientData,  

    FS\_LPCWSTR lpwsFunction,  

    FS\_LPCWSTR lpwsAction,  

    FS\_LPCWSTR lpwsContent,  

    FS\_INT32 nLevel  

);  
  

void (*FROnCollectBitmapData2)(  

    FS\_LPVOID clientData,  

    FS\_LPCWSTR lpwsFunction,  

    FS\_LPCWSTR lpwsAction,  

    FS\_DIBitmap bitmap,  

    FS\_INT32 nLevel  

);  
  

} FR_DataCollectionHandlerCallbacksRec,  

*FR_DataCollectionHandlerCallbacks;
```

**IStructSize**

The size of data structure. It must be set to *sizeof( [FR\\_DataCollectionHandlerCallbacksRec](#) )*.

**clientData**

The user-supplied data.

**FR\_DocPropertyPageCallbacksRec**

A callback set for document-level property dialog. Using this callback set, a new property page can be attached to the document-level property dialog, and the event occurred when user click the *OK* button is sent to client. See [FRAppRegisterDocPropertyPageHandler](#) , [FRAppAddDocPropertyPage](#) .

**Syntax**

```

typedef struct _fr_docpropertypage_callbacks_{  

    unsigned long IStructSize;  

    FS\_LPVOID clientData;  

    void (*FRDocPropertyPageOnCreate)(  

        FS\_LPVOID clientData,  

        HWND window  

    );  
  

    void (*FRDocPropertyPageOnDestroy)(  

        FS\_LPVOID clientData  

    );  

}
```

```

void (*FRDocPropertyPageOnSaveData)(
    FS\_LPVOID clientData
);

} FR_DocPropertyPageCallbacksRec,
*FR_DocPropertyPageCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( [FR\\_DocPropertyPageCallbacksRec](#) )*.

**clientData**

The user-supplied data.

## **FR\_DRMCryptoCallbacksRec**

The structure containing the crypto handler callback functions. They are for PDF cryptographic operations (encryption and decryption). The crypto handler works with security handler which provides algorithm and key info.

### Syntax

```

typedef struct _fr_cdrm_crypto_callbacks_ {
    unsigned long IStructSize;
    FS\_LPVOID clientData;
    void (*FRCryptoSetKey)(
        FS\_LPVOID clientData,
        FS\_KeyType ptype,
        FS\_LPBYTE bsKey,
        FS\_INT32 pnKeyLen
    );

    void (*FRCryptoCurrentEncryptObjNum)(
        FS\_LPVOID clientData,
        FS\_DWORD dwObjNum
    );

} FR_DRMCryptoCallbacksRec, *FR_DRMCryptoCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( [FR\\_CryptoCallbacksRec](#) )*.

**clientData**

The user-supplied data.

## **FR\_DRMSecurityCallbacksRec**

The structure containing the cDRM security handler callback functions. Plug-ins can process the PDF documents that are encrypted by customer security handler through these callbacks. See [FRAppRegisterDRMSecurityHandler](#) .

## Syntax

```
typedef struct _fr_cdrm_securitycallbacks_ {
    unsigned long IStructSize;
    FS_LPVVOID clientData;
    FS_LPVVOID (*FRSecurityOnInit)(
        FS_LPVVOID clientData,
        FS_LPCWSTR filePath,
        FPD_Object encryptDict,
        FPD_Document doc
    );
    FS_LPVVOID (*FRSecurityGetPermissions)(
        FS_LPVVOID clientData,
        FS_LPVVOID securityData,
        clientData,
        securityData
    );
    FS_BOOL (*FRSecurityIsProcessErrMsg)(
        FS_LPVVOID clientData
    );
} FR_DRMSecurityCallbacksRec, *FR_DRMSecurityCallbacks;
```

### IStructSize

The size of data structure. It must be set to *sizeof* ([FR\\_DRMSecurityCallbacks](#) ).

### clientData

The user-supplied data.

## FR\_EmptyFramWndNotifiesRec

A callback set for an empty frame window which created by [FRAppCreateAnEmptyFrameWnd](#) ().

## Syntax

```
typedef struct _fr_emptyframwndnotifies_ {
    unsigned long IStructSize;
    FS_LPVVOID clientData;
    FS_BOOL (*FRAppEmptyFramWndCanClose)(
        FS_LPVVOID clientData,
        HWND wnd
    );
} FR_EmptyFramWndNotifiesRec, *FR_EmptyFramWndNotifies;
```

---

} **FR\_EmptyFramWndNotifiesRec**, \***FR\_EmptyFramWndNotifies**;

**IStructSize**

The size of data structure. It must be set to *sizeof( FR\_EmptyFramWndNotifiesRec )*.

**clientData**

The user-supplied data.

**FR\_ExtraPrintInfoProviderCallbackRec**

You can use this interface to customize the print process. See [FRAppRegisterExtraPrintInfoProvider](#) .

**Syntax**

```
typedef struct _fr_print_extrainfo_provider_callbacks_{
    unsigned long IStructSize;
    FS_LPVVOID clientData;
    FS_BOOL (*FRTоМаст)(  
        FS_LPVVOID clientData,  
        FR_Document frDoc  
    );  
  
    FS_BOOL (*FRCCanBePrinted)(  
        FS_LPVVOID clientData,  
        FR_Document frDoc,  
        FS_INT32 copies,  
        FS_BOOL bTotalPages,  
        FS_DWordArray arrPages,  
        FS_BOOL bPostScript,  
        FS_BOOL bLocal,  
        FS_BOOL bNetWork,  
        FS_BOOL bShared,  
        FS_BytString Printer,  
        FS_BytString PrinterModel,  
        FS_BytString PrinterDriver,  
        FS_BytString PrinterData,  
        FS_BytString PrinterProt  
    );  
  
    FS_INT32 (*FRCCountOwnerTextPrintInfo)(  
        FS_LPVVOID clientData,  
        FR_Document frDoc,  
        FS_INT32 iPage  
    );  
  
    FS_INT32 (*FRCCountOwnerImgPrintInfo)(  
        FS_LPVVOID clientData,  
        FR_Document frDoc,  
        FS_INT32 iPage  
    );
```

```
FS_ByteString (*FRPrintOwnerTextInfo)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 iPage,  
    FS_INT32 nStartX,  
    FS_INT32 nStartY,  
    FS_INT32 nWidth,  
    FS_INT32 nHeight,  
    FS_INT32 index,  
    FS_INT32* textLeftPos,  
    FS_INT32* textTopPos,  
    FS_FLOAT* textOpacity,  
    FS_DWORD* textColor,  
    void* textUseFont,  
    FS_AffineMatrix* textMatrix  
);  
  
FS_DIBitmap (*FRPrintOwnerImgInfo)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 iPage,  
    FS_INT32 nStartX,  
    FS_INT32 nStartY,  
    FS_INT32 nWidth,  
    FS_INT32 nHeight,  
    FS_INT32 index,  
    FS_INT32* imgLeftPos,  
    FS_INT32* imgTopPos,  
    FS_FLOAT* imgOpacity,  
    FS_AffineMatrix* imgMatrix  
);  
  
void (*FRPrintOnPreviewDidRender)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FPD_Page pdfPage,  
    FS_INT32 iPage,  
    FS_AffineMatrix matrix,  
    FPD_RenderDevice renderDevice  
);  
  
void (*FRPrintOnDidRender)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FPD_Page pdfPage,  
    FS_INT32 iPage,  
    FS_AffineMatrix matrix,  
    FPD_RenderDevice renderDevice  
);  
  
FS_BOOL (*FRPrintIsPrintOpacity)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FPD_Page pdfPage,  
    FS_INT32 iPage  
);  
  
FS_BOOL (*FRCCanBePrinted2)(
```

```

    FS_LPVOID clientData,
    FR_Document frDoc,
    FS_INT32 copies,
    FS_BOOL bTotalPages,
    FS_DWordArray arrPages,
    FS_BOOL bPostScript,
    FS_BOOL bLocal,
    FS_BOOL bNetWork,
    FS_BOOL bShared,
    FS_WideString Printer,
    FS_BytString PrinterModel,
    FS_BytString bsSubset,
    FS_BytString PrinterDriver,
    FS_BytString PrinterData,
    FS_BytString PrinterProt
);

} FR_ExtraPrintInfoProviderCallbackRec,
*FR_ExtraPrintInfoProviderCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( [FR\\_ExtraPrintInfoProviderCallbackRec](#) )*.

**clientData**

The user-supplied data.

## [FR\\_MousePtCallbacksRec](#)

A callback set for mouse point handler. It is used to deal with the mouse operations. For example, you can receive mouse events. It can be associated to the tool by [FRTToolSetAssociatedMousePtHandlerType](#) . See [FRApplRegisterMousePtHandler](#) .

### Syntax

```

typedef struct _fr_mousept_callbacks{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_LPSTR (*FRCaptureGetType)(
        FS_LPVOID clientData
    );

    void* (*FRMousePtGetObjectAtPoint)(
        FS_LPVOID clientData,
        FR_PageView pageView,
        FS_DevicePoint point
    );

    FS_BOOL (*FRMousePtLButtonDown)(
        FS_LPVOID clientData,
        FR_PageView pageView,
        void* curData,
        unsigned int nFlags,

```

FS\_DevicePoint point  
);

FS\_BOOL (\*FRMousePtLButtonUp)(  
    FS\_LPVOID clientData,  
    FR\_PageView pageView,  
    void\* curData,  
    unsigned int nFlags,  
    FS\_DevicePoint point  
);

FS\_BOOL (\*FRMousePtLButtonDblClk)(  
    FS\_LPVOID clientData,  
    FR\_PageView pageView,  
    void\* curData,  
    unsigned int nFlags,  
    FS\_DevicePoint point  
);

FS\_BOOL (\*FRMousePtMouseMove)(  
    FS\_LPVOID clientData,  
    FR\_PageView pageView,  
    void\* curData,  
    unsigned int nFlags,  
    FS\_DevicePoint point  
);

FS\_BOOL (\*FRMousePtRButtonDown)(  
    FS\_LPVOID clientData,  
    FR\_PageView pageView,  
    void\* curData,  
    unsigned int nFlags,  
    FS\_DevicePoint point  
);

FS\_BOOL (\*FRMousePtRButtonUp)(  
    FS\_LPVOID clientData,  
    FR\_PageView pageView,  
    void\* curData,  
    unsigned int nFlags,  
    FS\_DevicePoint point  
);

FS\_BOOL (\*FRMousePtRButtonDblClk)(  
    FS\_LPVOID clientData,  
    FR\_PageView pageView,  
    void\* curData,  
    unsigned int nFlags,  
    FS\_DevicePoint point  
);

FS\_BOOL (\*FRMousePtMouseWheel)(  
    FS\_LPVOID clientData,  
    FR\_PageView pageView,  
    void\* curData,  
    unsigned int nFlags,  
    short zDelta,  
    FS\_DevicePoint point  
);

```

void (*FRMousePtOnMouseEnter)(  

    FS\_LPVOID clientData,  

    FR\_PageView pageView,  

    void* curData  

);  
  

void (*FRMousePtOnMouseExit)(  

    FS\_LPVOID clientData,  

    FR\_PageView pageView,  

    void* curData  

);  
  

} FR_MousePtCallbacksRec, *FR_MousePtCallbacks;
```

**IStructSize**

The size of data structure. It must be set to *sizeof( [FR\\_MousePtCallbacksRec](#) )*.

**clientData**

The user-supplied data.

## **FR\_OwnerFileTypeHandlerCallbacksRec**

The structure containing owner file type handler callbacks. You can control the process of owner file type by implementing the callbacks, such as opening, saving, sending email and so on. You can register it by [FRAppRegisterOwnerFileTypeIII](#) .

### Syntax

```

typedef struct _fr_ownerfiletype_handler_callbacks_{  

    unsigned long IStructSize;  

    FS\_LPVOID clientData;  

    FS\_BOOL (*FROwnerFileTypeHandlerDoOpen)(  

        FS\_LPVOID clientData,  

        FS\_LPCSTR lpszFilterName,  

        FS\_LPCWSTR lpszPath  

    );  
  

    FS\_BOOL (*FROwnerFileTypeHandlerDoSave)(  

        FS\_LPVOID clientData,  

        FS\_LPCSTR lpszFilterName,  

        FS\_LPCWSTR lpszPath  

    );  
  

    FS\_BOOL (*FROwnerFileTypeHandlerDoEmail)(  

        FS\_LPVOID clientData,  

        FS\_LPCSTR lpszFilterName,  

        FS\_LPCWSTR lpszPath  

    );  
  

    FS\_BOOL (*FROwnerFileTypeHandlerCanSetFileAssociation)(  

        FS\_LPVOID clientData,
```

```
FS_LPCWSTR lpszFileExt
);

FS_BOOL (*FROwnerFileTypeHandlerSetFileAssociationInfo)(
    FS_LPVOID clientData,
    FS_LPCWSTR lpszFileExt,
    FS_WideString wsRegClass,
    FS_WideString wsDesc,
    FS_WideString wsAppPath,
    FS_INT32* nDefaultIcon
);

FS_INT32 (*FROwnerFileTypeHandlerGetSupportFileTypeCount)(
    FS_LPVOID clientData
);

FS_BOOL (*FROwnerFileTypeHandlerGetFileFilterNameByIndex)(
    FS_LPVOID clientData,
    FS_INT32 nIndex,
    FS_BytString bsName
);

FS_BOOL (*FROwnerFileTypeHandlerGetFileTypeFilter)(
    FS_LPVOID clientData,
    FS_LPCSTR lpszFilterName,
    FS_WideString wsName
);

FS_BOOL (*FROwnerFileTypeHandlerGetFileExt)(
    FS_LPVOID clientData,
    FS_LPCSTR lpszFilterName,
    FS_WideString wsFileExt
);

FS_BOOL (*FROwnerFileTypeHandlerIsEnableSaveAsSettingBtn)(
    FS_LPVOID clientData,
    FS_LPCSTR lpszFilterName
);

FS_BOOL (*FROwnerFileTypeHandlerExecuteSaveAsSetting)(
    FS_LPVOID clientData,
    FS_LPCSTR lpszFilterName,
    HWND hParentWnd
);

FS_BOOL (*FROwnerFileTypeHandlerCanSupportAddToRecentList)(
    FS_LPVOID clientData
);

FS_BOOL (*FROwnerFileTypeHandlerCanSupportSave)(
    FS_LPVOID clientData,
    FS_WideString wsFilterName,
    FR_Document frDoc
);

FS_BOOL (*FROwnerFileTypeHandlerCanBeSaveAsToOtherExt)(
    FS_LPVOID clientData,
    FS_WideString wsFilterName
);
```

```

);

FS\_BOOL (*FROwnerFileTypeHandlerCanSupportOpen)(
    FS\_LPVOID clientData,
    FS\_WideString wsFilterName,
    FS\_WideString wsPathName
);

FS\_BOOL (*FROwnerFileTypeHandlerDoOpenDir)(
    FS\_LPVOID clientData,
    HWND csHandler
);

FS\_BOOL (*FROwnerFileTypeHandlerCanSupportOpenDir)(
    FS\_LPVOID clientData,
    HWND csHandler
);

FS\_BOOL (*FROwnerFileTypeHandlerSaveFilePathToClipboard)(
    FS\_LPVOID clientData,
    HWND csHandler
);

} FR_OwnerFileTypeHandlerCallbacksRec,
*FR_OwnerFileTypeHandlerCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FR\_OwnerFileTypeHandlerCallbacksRec )*.

**clientData**

The user-supplied data.

**FR\_PageEventCallbacksRec**

A callback set for document-level event handler. See  
[FRAppRegisterDocHandlerOfPDDoc](#) .

**Syntax**

```

typedef struct _fr_docevent_callbacks_{
    unsigned long IStructSize;
    FS\_LPVOID clientData;
    void (*FRDocWillOpen)(
        FS\_LPVOID clientData,
        FR\_Document doc
    );

    void (*FRDocDidOpen)(
        FS\_LPVOID clientData,
        FR\_Document doc
    );
}

```

```
void (*FRDocOnActivate)(  
    FS_LPVOID clientData,  
    FR_Document doc  
);  
  
void (*FRDocOnDeactivate)(  
    FS_LPVOID clientData,  
    FR_Document doc  
);  
  
void (*FRDocWillSave)(  
    FS_LPVOID clientData,  
    FR_Document doc,  
    FS_BOOL bSaveAs  
);  
  
void (*FRDocDidSave)(  
    FS_LPVOID clientData,  
    FR_Document doc,  
    FS_BOOL bSaveAs  
);  
  
void (*FRDocWillClose)(  
    FS_LPVOID clientData,  
    FR_Document doc  
);  
  
void (*FRDocDidClose)(  
    FS_LPVOID clientData,  
    FR_Document doc  
);  
  
void (*FRDocDidCopy)(  
    FS_LPVOID clientData,  
    FR_Document doc  
);  
  
void (*FRDocWillPrint)(  
    FS_LPVOID clientData,  
    FR_Document doc  
);  
  
void (*FRDocDidPrint)(  
    FS_LPVOID clientData,  
    FR_Document doc  
);  
  
void (*FRDocOnChange)(  
    FS_LPVOID clientData,  
    FR_Document doc  
);  
  
void (*FRDocOnPermissionChange)(  
    FS_LPVOID clientData,  
    FR_Document doc  
);
```

```
void (*FRDocWillDraw)(  
    FS_LPVOID clientData,  
    FR_DocView docView,  
    HDC dc  
);  
  
void (*FRDocDidDraw)(  
    FS_LPVOID clientData,  
    FR_DocView docView,  
    HDC dc  
);  
  
void (*FRDocOnWillInsertPages)(  
    FS_LPVOID clientData,  
    FR_Document doc,  
    FS_INT32 iInsertAt,  
    FS_INT32 nPages  
);  
  
void (*FRDocOnDidInsertPages)(  
    FS_LPVOID clientData,  
    FR_Document doc,  
    FS_INT32 iInsertAt,  
    FS_INT32 nPages  
);  
  
void (*FRDocOnWillDeletePages)(  
    FS_LPVOID clientData,  
    FR_Document doc,  
    FS_WordArray arrDelPages  
);  
  
void (*DidDeletePages)(  
    FS_LPVOID clientData,  
    FR_Document doc,  
    FS_WordArray arrDelPages  
);  
  
void (*FRDocOnWillModifyPageAttribute)(  
    FS_LPVOID clientData,  
    FR_Document doc,  
    FS_INT32 iPage  
);  
  
void (*FRDocOnDidModifyPageAttribute)(  
    FS_LPVOID clientData,  
    FR_Document doc,  
    FS_INT32 iPage  
);  
  
void (*FRDocOnWindowCreate)(  
    FS_LPVOID clientData,  
    FR_Document doc  
);  
  
void (*FRDocOnWindowDestroy)(  
    FS_LPVOID clientData,  
    FR_Document doc
```

```
);

void (*FRDocOnFrameCreate)(
    FS_LPVOID clientData,
    FR_Document doc,
    HWND hFrameWnd
);

void (*FRDocOnFrameDestroy)(
    FS_LPVOID clientData,
    FR_Document doc,
    HWND hFrameWnd
);

void (*FRDocOnAnnotSelectionChanged)(
    FS_LPVOID clientData
);

void (*FRDocOnAutoScrollBegin)(
    FS_LPVOID clientData,
    FR_DocView docView
);

void (*FRDocOnAutoScrollEnd)(
    FS_LPVOID clientData,
    FR_DocView docView
);

void (*FRDocOnFinishRender)(
    FS_LPVOID clientData,
    FR_DocView docView
);

void (*FRDocThumbnailWillDraw)(
    FS_LPVOID clientData,
    FR_ThumbnailView thumbnailView,
    HDC dc
);

void (*FRDocThumbnailDidDraw)(
    FS_LPVOID clientData,
    FR_ThumbnailView thumbnailView,
    HDC dc
);

void (*FRDocScrollBarThumbnailViewWillDraw)(
    FS_LPVOID clientData,
    FR_ScrollBarThumbnailView thumbnailView,
    HDC dc
);

void (*FRDocDidFileClose)(
    FS_LPVOID clientData,
    FS_LPCWSTR lpwsFilePath
);

FS_BOOL (*FRDocCanBeSaved)(
    FS_LPVOID clientData,
```

```
    FR_Document doc  
);  
  
    FS_BOOL (*OnDocPromptToSave)(  
        FS_LPVOID clientData,  
        FR_Document doc,  
        FS_BOOL* bCancel  
);  
  
    void (*FRDocWillReOpen)(  
        FS_LPVOID clientData,  
        FR_Document doc,  
        FS_BOOL bMemDoc  
);  
  
    void (*FRDocDidReOpen)(  
        FS_LPVOID clientData,  
        FR_Document doc,  
        FS_BOOL bMemDoc  
);  
  
    void (*FRDocOnFrameSize)(  
        FS_LPVOID clientData,  
        FR_Document doc,  
        HWND hFrameWnd,  
        FS_Rect rcClient  
);  
  
    void (*FRDocOnWillActivate)(  
        FS_LPVOID clientData,  
        FR_Document doc  
);  
  
    void (*FRDocOnWillDeactivate)(  
        FS_LPVOID clientData,  
        FR_Document doc  
);  
  
    void (*FRDocOnOtherDocActivate)(  
        FS_LPVOID clientData  
);  
  
    void (*FRDocOnOtherDocDeactivate)(  
        FS_LPVOID clientData  
);  
  
    void (*FRDocOnOtherDocClose)(  
        FS_LPVOID clientData  
);  
  
    void (*FRDocDidSave2)(  
        FS_LPVOID clientData,  
        FR_Document doc,  
        FS_BOOL bSaveAs,  
        FS_BOOL bPromptToSave  
);  
  
    FS_BOOL (*FRDocOnKeyDown)(
```

```
FS_LPVOID clientData,
FR_DocView docView,
unsigned int nKeyCode,
unsigned int nFlags
);

FS_BOOL (*FRDocOnKeyUp)(  
    FS_LPVOID clientData,  
    FR_DocView docView,  
    unsigned int nKeyCode,  
    unsigned int nFlags  
);

FS_BOOL (*FRDocOnChar)(  
    FS_LPVOID clientData,  
    FR_DocView docView,  
    unsigned int nKeyCode,  
    unsigned int nFlags  
);

FS_BOOL (*FRDocOnLButtonDown)(  
    FS_LPVOID clientData,  
    FR_PageView pageView,  
    unsigned int nFlags,  
    FS_DevicePoint point  
);

FS_BOOL (*FRDocOnLButtonUp)(  
    FS_LPVOID clientData,  
    FR_PageView pageView,  
    unsigned int nFlags,  
    FS_DevicePoint point  
);

FS_BOOL (*FRDocOnLButtonDblClk)(  
    FS_LPVOID clientData,  
    FR_PageView pageView,  
    unsigned int nFlags,  
    FS_DevicePoint point  
);

FS_BOOL (*FRDocOnMouseMove)(  
    FS_LPVOID clientData,  
    FR_PageView pageView,  
    unsigned int nFlags,  
    FS_DevicePoint point  
);

FS_BOOL (*FRDocOnRButtonDown)(  
    FS_LPVOID clientData,  
    FR_PageView pageView,  
    unsigned int nFlags,  
    FS_DevicePoint point  
);

FS_BOOL (*FRDocOnRButtonUp)(  
    FS_LPVOID clientData,  
    FR_PageView pageView,  
    unsigned int nFlags,
```

```
FS_DevicePoint point
);

FS_BOOL (*FRDocOnRButtonDblClk)(
    FS_LPVOID clientData,
    FR_PageView pageView,
    unsigned int nFlags,
    FS_DevicePoint point
);

FS_BOOL (*FRDocOnMouseWheel)(
    FS_LPVOID clientData,
    FR_PageView pageView,
    unsigned int nFlags,
    short zDelta,
    FS_DevicePoint point
);

void (*FRDocOnDrawAnnot)(
    FS_LPVOID clientData,
    FR_PageView pageView,
    FR_Annot frAnnot,
    HDC hDC,
    FPD_RenderDevice renderDevice,
    FS_AffineMatrix user2Device,
    FS_Rect rcWindow
);

void (*FRDocOnDocCollectActionData)(
    FS_LPVOID clientData,
    FR_Document doc,
    FS_LPCWSTR lpwsOperatorType,
    FS_LPCWSTR lpwsOperator,
    FS_MapPtrToPtr valueMap
);

void (*FRDocWillOpen2)(
    FS_LPVOID clientData,
    FR_Document doc,
    FS_LPCWSTR lpwsFilePath
);

void (*FRDocOnOptimizerFinish)(
    FS_LPVOID clientData,
    FR_Document doc
);

FS_BOOL (*FRDocCanBeClose)(
    FS_LPVOID clientData,
    FR_Document doc
);

void (*FRDocOnReOpenFailed)(
    FS_LPVOID clientData,
    FS_LPCWSTR lpwsFilePath
);

FS_BOOL (*FRDocWillSave2)(
```

```
FS_LPVOID clientData,
FR_Document doc,
FS_BOOL bSaveAs,
FS_LPCWSTR lpwsFilePath,
clientData,
FS_BOOL bCanSupportPDFOnly,
FS_BOOL bChoice,
FS_LPCWSTR* pwszFilePath,
FS_INT32 iIndex
);

FS_BOOL (*FRDocSaveAsBeforeReopen)(
    FS_LPVOID clientData,
    FS_LPCWSTR wsFileName
);

FS_BOOL (*FRDocCanPaste)(
    FS_LPVOID clientData
);

void (*OnMouseClickOnText)(
    FS_LPVOID clientData,
    FR_Document doc,
    FS_LPCWSTR wsText,
    FS_Rect rect
);

BOOL (*FRDocOwnerSaveAs)(
    FS_LPVOID clientData,
    FR_Document doc,
    FS_LPCWSTR wszPathName
);

FS_BOOL (*FRDocOnCanDetache)(
    FS_LPVOID clientData,
    FR_Document doc
);

void (*FRDocDelayDidOpen)(
    FR_Document doc
);

void (*FRDocOnActivate2)(
    FS_LPVOID clientData,
    FR_Document doc,
    FS_BOOL bMainfrmActivating
);

void (*FRDocOnDeactivate2)(
    FS_LPVOID clientData,
    FR_Document doc,
    FS_BOOL bMainfrmActivating,
    clientData,
    bMainfrmActivating,
    clientData,
    doc,
    FR_Annot focusAnnot,
    clientData,
    FR_PageView pv
```

```

);
void (*FRPageViewOnClose)(  

    FS\_LPVOID clientData,  

    FR\_PageView pv  

);
void (*FRPageViewOnVisible)(  

    FS\_LPVOID clientData,  

    FR\_PageView pv  

);
void (*FRPageViewInvisible)(  

    FS\_LPVOID clientData,  

    FR\_PageView pv  

);
void (*FRPageViewOnContentChanged)(  

    FS\_LPVOID clientData,  

    FR\_PageView pv  

);
void (*FRPageViewOnWillParsePage)(  

    FS\_LPVOID clientData,  

    FR\_PageView pv,  

    FS\_BOOL bPageVisible  

);
void (*FRPageViewOnDidParsePage)(  

    FS\_LPVOID clientData,  

    FR\_PageView pv,  

    FS\_BOOL bPageVisible  

);
void (*FRPageViewOnContentChanged2)(  

    FS\_LPVOID clientData,  

    FR\_PageView pv,  

    FS\_PtrArray objArray,  

    FR\_ContentChangeType changeType  

);
} FR_PageEventCallbacksRec, *FR_PageEventCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( [FR\\_DocEventCallbacksRec](#) )*.

**clientData**

The user-supplied data.

**FR\_PanelViewCallbacksRec**

A callback set for navigation panel view. A navigation panel is a container that contain the navigation views, such as bookmark view, thumbnail pages view, layers view, comments view, attachments view and so on. See [FRAppRegisterNavPanelView](#) .

## Syntax

```
typedef struct _fr_panelview_callbacks_{
    unsigned long lStructSize;
    FS_LPVVOID clientData;
    FS_LPSTR (*FRPanelViewGetName)(
        FS_LPVVOID clientData
    );

    FS_LPWSTR (*FRPanelViewGetTitle)(
        FS_LPVVOID clientData
    );

    void (*FRPanelViewInitNewView)(
        FS_LPVVOID clientData,
        FPD_Document doc,
        HWND window
    );

    HWND (*FRPanelViewOnPanelActive)(
        FS_LPVVOID clientData
    );

    void (*FRPanelViewOnActive)(
        FS_LPVVOID clientData,
        FPD_Document doc,
        HWND window
    );

    void (*FRPanelViewOnRotate)(
        FS_LPVVOID clientData,
        FPD_Document doc,
        FS_INT32 nRotate
    );

    void (*FRPanelViewOnDestroyView)(
        FS_LPVVOID clientData,
        HWND window,
        FPD_Document doc
    );

    FS_BOOL (*FRPanelViewIsDockToBottom)(
        FS_LPVVOID clientData
    );

    void (*FRPanelViewGetButtonTip)(
        FS_LPVVOID clientData,
        FS_WideString* csOutTip
    );

    void (*FRPanelViewGetButtonDescriptText)(
        FS_LPVVOID clientData,
        FS_WideString* csOutText
    );
};
```

---

```

FS_DIBitmap (*FRPanelViewGetButtonIcon)(
    FS_LPVOID clientData
);

void (*FRPanelViewSetPos)(
    FS_LPVOID clientData,
    FPD_Document doc,
    FS_INT32 nPage
);

} FR_PanelViewCallbacksRec, *FR_PanelViewCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FR\_PanelViewCallbacksRec )*.

**clientData**

The user-supplied data.

**FR\_PDFAPPluginHandlerCallbacksRec****Syntax**

```

typedef struct _fr_pdafapluginhandler_callbacks_{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_BOOL (*FRPDFA_SaveAsPDFA)(
        FS_LPVOID clientData,
        FR_Document pRDoc,
        const FRPDFA_PDFVersion pVersion,
        FS_WideString wsPathSuffix
    );
}

} FR_PDFAPPluginHandlerCallbacksRec,
*FR_PDFAPPluginHandlerCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FR\_PDFAPPluginHandlerCallbacksRec )*.

**clientData**

The user-supplied data.

**FR\_POEventCallbacksRec**

A callback set for page organizing event handler. These callbacks will be invoked when Foxit Reader does the page organizing, such as adding pages, deleting pages, replacing pages and so on. See [FRAppRegisterPOEventHandler](#) .

## Syntax

```
typedef struct _fr_poevent_callbacks_{
    unsigned long lStructSize;
    FS_LPVOID clientData;
    void (*FRPOOnBeforeInsertPages)(
        FS_LPVOID clientData,
        FR_Document frDoc,
        FS_INT32 nInsertAt,
        FS_INT32 nCount
    );

    void (*FRPOOnDoInsertPagesDictFinish)(
        FS_LPVOID clientData,
        FPD_Document fpdDesDoc,
        FS_INT32 nInsertAt,
        FPD_Document fpdSrcDoc,
        FS_WordArray arrSrcPages,
        FS_BOOL bEntireDoc
    );

    void (*FRPOOnAfterInsertPages)(
        FS_LPVOID clientData,
        FR_Document frDoc,
        FS_INT32 nInsertAt,
        FS_INT32 nCount
    );

    void (*FRPOOnBeforeDeletePages)(
        FS_LPVOID clientData,
        FR_Document frDoc,
        FS_WordArray arrDelPages
    );

    void (*FRPOOnAfterDeletePages)(
        FS_LPVOID clientData,
        FR_Document frDoc,
        FS_WordArray arrDelPages
    );

    void (*FRPOOnBeforeReplacePages)(
        FS_LPVOID clientData,
        FR_Document frDoc,
        FS_INT32 nStart,
        FPD_Document fpdSrcDoc,
        FS_WordArray arrSrcPages
    );

    void (*FRPOOnAfterReplacePages)(
        FS_LPVOID clientData,
        FR_Document frDoc,
        FS_INT32 nStart,
        FPD_Document fpdSrcDoc,
        FS_WordArray arrSrcPages
    );
}
```

```
void (*FRPOOnBeforeSwapTwoPage)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 iPage1,  
    FS_INT32 iPage2  
);  
  
void (*FRPOOnAfterSwapTwoPage)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 iPage1,  
    FS_INT32 iPage2  
);  
  
void (*FRPOOnBeforeRotatePage)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 iPage,  
    FS_INT32 nRotate  
);  
  
void (*FRPOOnAfterRotatePage)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 iPage,  
    FS_INT32 nRotate  
);  
  
void (*FRPOOnBeforeResizePage)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 iPage,  
    FS_FloatRect MediaBox,  
    FS_FloatRect CropBox  
);  
  
void (*FRPOOnAfterResizePage)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 iPage,  
    FS_FloatRect MediaBox,  
    FS_FloatRect CropBox  
);  
  
void (*FRPOOnBeforeExtractPage)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FS_WordArray arrSrcPages,  
    FPD_Document fpdDstDoc  
);  
  
void (*FRPOOnAfterExtractPage)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FS_WordArray arrSrcPages,  
    FPD_Document fpdDstDoc  
);  
  
void (*FRPOOnBeforeModifyPageAttr)(
```

```

    FS_LPVOID clientData,
    FR_Document frDoc,
    FS_INT32 nStart,
    FS_INT32 nCount
);

void (*FRPOOnAfterModifyPageAttr)(
    FS_LPVOID clientData,
    FR_Document frDoc,
    FS_INT32 nStart,
    FS_INT32 nCount
);

void (*FRPOOnBeforeMovePages)(
    FS_LPVOID clientData,
    FR_Document frDoc,
    FS_INT32 nMoveTo,
    FS_WordArray ArrToMove
);

void (*FRPOOnAfterMovePages)(
    FS_LPVOID clientData,
    FR_Document frDoc,
    FS_INT32 nMoveTo,
    FS_WordArray ArrToMove
);

void (*FRPOOnRelease)(
    FS_LPVOID clientData
);
}

} FR_POEventCallbacksRec, *FR_POEventCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( [FR\\_POEventCallbacksRec](#) )*.

**clientData**

The user-supplied data.

**FR\_PreferPageCallbacksRec**

A callback set for preferences dialog. Using this callback set, a new preference page can be attached to the preferences dialog, and the event occurred when user click the *OK* button is sent to client. See [FRAppRegisterPreferencePageHandler](#) , [FRAppAddPreferencePage](#) .

**Syntax**

```

typedef struct _fr_preferencepage_callbacks{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    void (*FRPrefPageOnCreate)(

```

---

```

    FS_LPVOID clientData,
    HWND window
);

void (*FRPrefPageOnDestroy)(
    FS_LPVOID clientData
);

void (*FRPrefPageOnSaveData)(
    FS_LPVOID clientData
);

void (*FRPrefPageOnGetTabTitle)(
    FS_LPVOID clientData,
    FS_WideString wsTitle
);

} FR_PreferPageCallbacksRec, *FR_PreferPageCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof*( [FR\\_PreferPageCallbacksRec](#) ).

**clientData**

The user-supplied data.

**[FR\\_RibbonRecentFileEventCallbacksRec](#)**

A callback set for Ribbon recent file list event handler. These callbacks will be invoked when the item of the recent file list is pinned or removed. See [FRApplRegisterRibbonRecentFileEventHandler](#) .

**Syntax**

```

typedef struct _fr_ribbonrecentfileevent_callbacks_{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    void (*FRRibbonRecentFileChangItemPinned)(
        FS_LPVOID clientData,
        FS_LPCSTR lpcsRecentListName,
        FS_INT32 nIndex,
        FS_LPCWSTR lpwsFilePath,
        FS_INT32 nPinned
    );

    void (*FRRibbonRecentFileRemoveItem)(
        FS_LPVOID clientData,
        FS_LPCSTR lpcsRecentListName,
        FS_INT32 nIndex,
        FS_LPCWSTR lpwsFilePath
    );

```

```
 } FR_RibbonRecentFileEventCallbacksRec,
*FR_RibbonRecentFileEventCallbacks;
```

**IStructSize**

The size of data structure. It must be set to *sizeof* ([FR\\_RibbonRecentFileEventCallbacksRec](#)).

**clientData**

The user-supplied data.

## FR\_SecurityCallbacksRec

The structure containing the security handler callback functions. Plug-ins can process the PDF documents that are encrypted by customer security handler through these callbacks. See [FRAppRegisterSecurityHandler](#).

### Syntax

```
typedef struct _fr_securitycallbacks_{
    unsigned long IStructSize;
    FS_LPVVOID clientData;
    FS_LPVOID (*FRSecurityOnInit)(
        FS_LPVVOID clientData,
        FS_LPCWSTR filePath,
        FPD_Object encryptDict,
        FPD_Document doc
    );

    FS_BOOL (*FRSecurityIsProcessErrMsg)(
        FS_LPVVOID clientData
    );

    FS_DWORD (*FRSecurityGetPermissions)(
        FS_LPVVOID clientData,
        FS_LPVVOID securityData
    );

    FS_BOOL (*FRSecurityIsOwner)(
        FS_LPVVOID securityData,
        FS_LPVVOID clientData
    );

    FS_BOOL (*FRSecurityGetCryptInfo)(
        FS_LPVVOID clientData,
        FS_LPVVOID securityData,
        int* outCipher,
        FS_LPVOID* outBuffer,
        FS_INT32* outKeylen
    );

    FS_BOOL (*FRSecurityIsMetadataEncrypted)(
        FS_LPVVOID securityData,
```

---

```

FS_LPVOID clientData
);

void (*FRSecurityFinishHandler)(
FS_LPVOID clientData,
FS_LPVOID securityData
);

FS_LPVOID (*FRSecurityCreateCryptoHandler)(
FS_LPVOID clientData,
FS_LPVOID securityData,
FS_INT32* outType
);

} FR_SecurityCallbacksRec, *FR_SecurityCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FR\_SecurityCallbacksRec )*.

**clientData**

The user-supplied data.

**FR\_SecurityMethodCallbacksRec**

The structure containing security method callbacks. You can manage the security methods of Foxit Reader by implementing the callbacks. You can register it by [FRAppRegisterSecurityMethod](#) .

**Syntax**

```

typedef struct _fr_securitymethod_callbacks_{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_LPWSTR (*FRSecurityMethodGetName)(
        FS_LPVOID clientData
    );

    FS_LPWSTR (*FRSecurityMethodGetTitle)(
        FS_LPVOID clientData
    );

    FS_BOOL (*FRSecurityMethodIsMyMethod)(
        FS_LPVOID clientData,
        FR\_Document doc
    );

    FS_BOOL (*FRSecurityMethodCheckModuleLicense)(
        FS_LPVOID clientData
    );

    FS_BOOL (*FRSecurityMethodCanBeModified)(
        FS_LPVOID clientData
    );
}

```

```

);

void (*FRSecurityMethodDoSetting)(
    FS\_LPVOID clientData,
    HWND hWnd,
    FS\_BOOL* bSuc
);

FS\_BOOL (*FRSecurityMethodRemoveSecurityInfo)(
    FS\_LPVOID clientData
);

HWND (*FRSecurityMethodCreatePermSubDlg)(
    FS\_LPVOID clientData,
    HWND hParent
);

void (*FRSecurityMethodDestroyPermSubDlg)(
    FS\_LPVOID clientData,
    HWND hWnd
);

} FR_SecurityMethodCallbacksRec,
*FR_SecurityMethodCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( [FR\\_SecurityMethodCallbacksRec](#) )*.

**clientData**

The user-supplied data.

**FR\_SelectionCallbacksRec**

A callback set for selection handler. The callbacks implement the selection handler functions. For example, they can add an item to the selection, remove an item from the selection, or copy the current selection to the clipboard. See [FRAppRegisterSelectionHandler](#) .

**Syntax**

```

typedef struct _fr_selection_callbacks{
    unsigned long IStructSize;
    FS\_LPVOID clientData;
    FS\_LPSTR (*FRSelectionGetType)(
        FS\_LPVOID clientData
    );

    FS\_BOOL (*FRSelectionCanSelectAll)(
        FS\_LPVOID clientData,
        FR\_Document document,
        void* curSelectData
    );

```

```
void (*FRSelectionDoSelectAll)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData  
);  
  
FS_BOOL (*FRSelectionCanDelete)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData  
);  
  
void (*FRSelectionDoDelete)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData  
);  
  
FS_BOOL (*FRSelectionCanCopy)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData  
);  
  
void (*FRSelectionDoCopy)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData  
);  
  
FS_BOOL (*FRSelectionCanCut)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData  
);  
  
void (*FRSelectionDoCut)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData  
);  
  
FS_BOOL (*FRSelectionCanPaste)(  
    FS_LPVOID clientData,  
    FR_Document document  
);  
  
void (*FRSelectionDoPaste)(  
    FS_LPVOID clientData,  
    FR_Document document  
);  
  
void (*FRSelectionLosingSelection)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData  
);
```

```
void (*FRSelectionGettingSelection)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData  
);  
  
void* (*FRSelectionRemovedFromSelection)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData,  
    void* remData  
);  
  
void* (*FRSelectionAddedToSelection)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData,  
    void* addData  
);  
  
void (*FRSelectionShowSelection)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData  
);  
  
FS_BOOL (*FRSelectionKeyDown)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData,  
    unsigned int nKeyCode,  
    unsigned int nFlags  
);  
  
FS_BOOL (*FRSelectionKeyUp)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData,  
    unsigned int nKeyCode,  
    unsigned int nFlags  
);  
  
FS_BOOL (*FRSelectionKeyChar)(  
    FS_LPVOID clientData,  
    FR_Document document,  
    void* curSelectData,  
    unsigned int nChar,  
    unsigned int nFlags  
);  
  
FS_BOOL (*FRSelectionMouseWheel)(  
    FS_LPVOID clientData,  
    FR_PageView pageView,  
    void* curSelectData,  
    unsigned int nFlags,  
    const FS_DevicePoint point  
);
```

```

FS_BOOL (*FRSelectionCanDeselectAll)(  

    FS_LPVOID clientData,  

    FR_Document document,  

    void* curSelectData  

);  
  

void (*FRSelectionDoDeselectAll)(  

    FS_LPVOID clientData,  

    FR_Document document,  

    void* curSelectData  

);  
  

} FR_SelectionCallbacksRec, *FR_SelectionCallbacks;
```

**IStructSize**

The size of data structure. It must be set to *sizeof( [FR\\_SelectionCallbacksRec](#) )*.

**clientData**

The user-supplied data.

**FR\_SignatureHandlerCallbacksRec**

A callback set for signature handler.

**Syntax**

```

typedef struct _fr_signaturehandler_callbacks_ {  

    unsigned long IStructSize;  

    FS_LPVOID clientData;  

    char* (*FRSignatureHandlerGetName)(  

        FS_LPVOID clientData  

    );  
  

    FR_SG_HANDLE (*FRSignatureHandlerSignData)(  

        FS_LPVOID clientData,  

        const unsigned char* pData2BSigned,  

        unsigned long ulData2BSignedLen,  

        unsigned char** pPSignedData,  

        unsigned long* pulSignedDataLen  

    );  
  

    FR_SG_HANDLE (*FRSignatureHandlerTimeStampDate)(  

        FS_LPVOID clientData,  

        FR_SignatureTimestamp* pSgTimeStampDate  

    );  
  

    FR_SG_HANDLE (*FRSignatureHandlerVerifyData)(  

        FS_LPVOID clientData,  

        const unsigned char* pSignedData,  

        unsigned long ulSignedDataLen,  

        const unsigned char* pData2BSigned,
```

```

unsigned long ulDataT2BSignedLen,
FS\_DWORD* outVerifyState,
void** hCertContext
);

FR\_SG\_HANDLE (*FRSignatureHandlerShowStateUI)(
FS\_LPVOID clientData,
const FS\_DWORD nVerifyState,
void* pWnd,
void* hCertContext,
FR\_SignatureDictInfo* pSignDictInfo
);

FR\_SG\_HANDLE (*FRSignatureHandlerCanClear)(
FS\_LPVOID clientData,
void* hCertContext,
FS\_BOOL* bCanClear
);

FR\_SG\_HANDLE (*FRSignatureHandlerShowSignProperties)(
FS\_LPVOID clientData,
const unsigned char* pSignedData,
unsigned long ulSignedDataLen,
const unsigned char* pData2BSigned,
unsigned long ulDataT2BSignedLen,
void* pWnd,
FR\_SignatureDictInfo* pSignDictInfo,
void* hCertContext
);

void (*RSignatureHandlerReleaseSignData)(
unsigned char pPSignedData,
unsigned long pulSignedDataLen
);

} FR_SignatureHandlerCallbacksRec,
*FR_SignatureHandlerCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof* ([FR\\_SignatureHandlerCallbacksRec](#) ).

**clientData**

The user-supplied data.

**FR\_StatusBarWndExCallbacksRec**

The plug-in can add a window to the status bar in the specified location. The callback set is used to create the status bar window.

**Syntax**

```
typedef struct _fr_statusbarwndex_callbacks_ {
```

```

unsigned long IStructSize;
FS_LPVOID clientData;
void (*FRStatusBarWndExOnGetTooltip)(  

    FS_LPVOID clientData,  

    FS_WideString outTooltip  

);
HWND (*FRStatusBarWndExOnCreateWnd)(  

    FS_LPVOID clientData,  

    void* pParent  

);
void (*FRStatusBarWndExGetSize)(  

    FS_LPVOID clientData,  

    FS_INT32* cx,  

    FS_INT32* cy  

);
void (*FRStatusBarWndExOnSize)(  

    FS_LPVOID clientData,  

    FS_INT32 cx,  

    FS_INT32 cy  

);
} FR_StatusBarWndExCallbacksRec,  

*FR_StatusBarWndExCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FR\_StatusBarWndExCallbacksRec )*.

**clientData**

The user-supplied data.

**FR\_SubscriptionProviderCallbacksRec**

A callback set for subscription provider. See [FRAppSetSubscriptionProvider](#) .

**Syntax**

```

typedef struct _fr_subscriptionprovider_callbacks_ {
    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_BOOL (*FRShowSubscribeRibbonUI)(  

        FS_LPVOID clientData,  

        FRSubscriptionFlowName subWorkflowName  

    );
    FS_BOOL (*FRStartSubscribeWorkflow)(  

        FS_LPVOID clientData,  

        FRSubscriptionFlowName subWorkflowName,  

        FS_WideString outReturnValue  

    );
}

```

```

FS_BOOL (*FRIsLicenseRevoked)(
    FS_LPVOID clientData
);

FS_BOOL (*FRShowSubscribeFlash)(
    FS_LPVOID clientData
);

} FR_SubscriptionProviderCallbacksRec,
*FR_SubscriptionProviderCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof*( [FR\\_SubscriptionProviderCallbacksRec](#) ).

**clientData**

The user-supplied data.

**FR\_UndoRedoCallbacksRec**

A callback set for undo-redo. The framework manage the undo-redo items for the plug-ins. The framework calls these interface to otify the plug-ins when to implement the undo-redo operation, when to release the undo-redo items. See [FRAppAddUndoRedoItem](#) .

**Syntax**

```

typedef struct _fr_undoredo_callbacks_{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    void (*OnUndo)(
        FS_LPVOID clientData
    );

    void (*OnRedo)(
        FS_LPVOID clientData
    );

    void (*OnRelease)(
        FS_LPVOID clientData
    );
}

} FR_UndoRedoCallbacksRec, *FR_UndoRedoCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof*( [FR\\_UndoRedoCallbacksRec](#) ).

**clientData**

The user-supplied data.

## FR\_WinMSGCallbacksRec

A callback set for creating an task pane view. A task pane is a part of main window of Foxit Reader, it is shown at left area of main window. Using [FRAppRegisterTaskPaneView](#) () to register a task pane view and then call [FRAppShowTaskPane](#) () to show it.

### Syntax

```
typedef struct _fr_taskpanerview_callbacks_{
    unsigned long lStructSize;
    FS_LPVOID clientData;
    FS_LPSTR (*FRTaskPaneViewGetName)(FS_LPVOID clientData
                                         );
    FS_LPWSTR (*FRTaskPaneViewGetTitle)(FS_LPVOID clientData
                                         );
    FS_LPWSTR (*FRTaskPaneViewGetDropMenuItemText)(FS_LPVOID clientData
                                                   );
    FS_LPWSTR (*FRTaskPaneViewGetDropMenuItemTip)(FS_LPVOID clientData
                                                   );
    HWND (*FRTaskPaneViewCreateHwnd)(FS_LPVOID clientData,
                                       HWND hOwner
                                       );
    void (*FRTaskPaneViewDestroyHwnd)(FS_LPVOID clientData,
                                      HWND hOwner
                                      );
    FS_DIBitmap (*FRTaskPaneViewGetBigIcon)(FS_LPVOID clientData
                                              );
    FS_DIBitmap (*FRTaskPaneViewGetSmallIcon)(FS_LPVOID clientData
                                              );
    void (*FRTaskPaneViewOnActive)(FS_LPVOID clientData,
                                   HWND hOwner
                                   );
    void (*FRTaskPaneViewOnDestroy)(FS_LPVOID clientData
                                   );
```

```

void (*\_FRTTaskPaneViewOnShowTaskPane)(  

    FS\_LPVOID clientData,  

    HWND hOwner,  

    FS\_BOOL bShow  

);  
  

FS\_BOOL (*\_FRTTaskPaneViewCanClose)(  

    FS\_LPVOID clientData  

);  
  

FS\_BOOL (*\_FRTaskPanelViewIsShowInMenu)(  

    FS\_LPVOID clientData,  

    clientData,  

    void* pMsg  

);
} FR_WinMSGCallbacksRec, *FR_WinMSGCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof*( [FR\\_TaskPaneViewCallbackRec](#) ).

**clientData**

The user-supplied data.

**FR\_WndProviderCallbacksRec**

The callback set is used to create a window above the document view. There are already some windows above the document view, such as PDF view, text view, ruler view. You can register it by [FRAppRegisterWndProvider](#) , d unregister it by [FRAppUnRegisterWndProvider](#) .

**Syntax**

```

typedef struct _fr_wndprovider_callbacks_{  

    unsigned long IStructSize;  

    FS\_LPVOID clientData;  

    FS\_LPCSTR (*FRWndProviderGetName)(FS\_LPVOID clientData);  

    void (*FRWndProviderCreateViewWnd)(  

        FS\_LPVOID clientData,  

        FR\_Document frDoc,  

        HWND hParent  

    );  
  

    void (*FRWndProviderOnHScroll)(  

        FS\_LPVOID clientData,  

        FR\_Document frDoc,  

        unsigned int nSBCode,  

        unsigned int nPos,  

        void* pScrollBar  

    );

```

```
void (*FRWndProviderOnVScroll)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    unsigned int nSBCode,  
    unsigned int nPos,  
    void* pScrollBar  
);  
  
FS_BOOL (*FRWndProviderOnCmdMsg)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    unsigned int nID,  
    FS_INT32 nCode,  
    void* pExtra,  
    void* pHandlerInfo  
);  
  
void (*FRWndProviderMoveWindow)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 x,  
    FS_INT32 y,  
    FS_INT32 nWidth,  
    FS_INT32 nHeight,  
    FS_BOOL bRepaint  
);  
  
FS_BOOL (*FRWndProviderOnSetCursor)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    void* pWnd,  
    unsigned int nHitTest,  
    unsigned int message  
);  
  
void (*FRWndProviderZoomToPage)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    double dbScale,  
    FS_BOOL bUpdate  
);  
  
void (*FRWndProviderGotoPage)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FS_INT32 nPageIndex  
);  
  
void (*FRWndProviderShowWindow)(  
    FS_LPVOID clientData,  
    FR_Document frDoc,  
    FS_BOOL bShow  
);  
  
void (*FRWndProviderReInitScrollBar)(  
    FS_LPVOID clientData,  
    FR_Document frDoc  
);
```

```

FS_INT32 (*FRWndProviderGetPageIndex)(  

    FS_LPVOID clientData,  

    FR_Document frDoc  

);

void (*FRWndProviderInitScrollBar)(  

    FS_LPVOID clientData,  

    FR_Document frDoc,  

    void* pHScroll,  

    void* pVScroll  

);

void (*FRWndProviderOnSetFocus)(  

    FS_LPVOID clientData,  

    FR_Document frDoc,  

    void* pOldWnd  

);

FS_BOOL (*FRWndProviderOnMouseWheel)(  

    FS_LPVOID clientData,  

    FR_Document frDoc,  

    unsigned int nFlags,  

    short zDelta,  

    FS_INT32 x,  

    FS_INT32 y  

);

void (*FRWndProviderOnRelease)(  

    FS_LPVOID clientData  

);

} FR_WndProviderCallbacksRec, *FR_WndProviderCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FR\_WndProviderCallbacksRec )*.

**clientData**

The user-supplied data.

**clientData)**

The user-supplied data.

**FR\_ActionWizardHandlerCallbacksRec****Syntax**

```

typedef struct __FR_ActionWizardHandlerCallbacksRec__{  

    unsigned long IStructSize;  

    FS_LPVOID clientData;  

    void (*OnGetName)(FS_LPVOID clientData, FS_ByteString outName);

```

---

```

void (*OnGetCommandName)(FS_LPVOID clientData, FS_ByteString
outCmdName);
FS_INT32 (*OnGetPresetFlag)(FS_LPVOID clientData);
FS_BOOL (*OnSetting)(FS_LPVOID clientData, FS_XMLElement
pXML, HWND hWnd);
FS_BOOL (*OnExecute)(FS_LPVOID clientData, FS_XMLElement pXML,
FS_BOOL bPromptUser, FRActionWizardExcuteStatus* outStatus);
FS_BOOL (*OnGetTitle)(FS_LPVOID clientData, FS_ByteString bsName,
FS_WideString outTitle);
FS_BOOL (*OnNeedReopen)(FS_LPVOID clientData);
FS_BOOL (*OnNeedChangeExecuteDoc)(FS_LPVOID clientData);
FS_BOOL (*OnIsEnableExecute)(FS_LPVOID clientData);
void (*OnExecuteDone)(FS_LPVOID clientData);
} FR_ActionWizardHandlerCallbacksRec,
*FR_ActionWizardHandlerCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FR\_ActionWizardHandlerCallbacksRec )*.

**clientData**

The user-supplied data.

**outName)**

The user-supplied data.

**outCmdName)**

The user-supplied data.

**clientData)**

The user-supplied data.

**hWnd)**

The user-supplied data.

**outStatus)**

The user-supplied data.

**outTitle)**

The user-supplied data.

**clientData)**

The user-supplied data.

**clientData)**

The user-supplied data.

**clientData)**

The user-supplied data.

**clientData)**

The user-supplied data.

## FR\_CloudLoginProviderCallbacksRec

A callback set for the service provider of cloud login. See [FRCLOUDLOGINPROVIDERSET](#).

### Syntax

```
typedef struct _fr_cloudloginprovider_callbacks_{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_BOOL (*FRCLOUDLOGINPROVIDERGetWebServiceURL)(FS_LPVOID
        clientData,FS_ByteString bsType,FS_WideString wsOutUrl);
    FS_BOOL (*FRCLOUDLOGINPROVIDERUploadImage)(FS_LPVOID clientData,
        FS_ByteString bsUserToken, FS_ByteString bsType,FS_ByteString
        bsContent,FS_WideString wsOutUrl);
    FS_BOOL (*FRCLOUDLOGINPROVIDEROnIsLogIn)(
        fwsHost,
        FS_LPVOID clientData
    );

    FS_BOOL (*FRCLOUDLOGINPROVIDEROnSignIn)(
        fwsHost,
        FS_LPVOID clientData
    );

    FS_BOOL (*FRCLOUDLOGINPROVIDEROnSignOut)(
        fwsDomain,
        FS_LPVOID clientData
    );

    FS_BOOL (*FRCLOUDLOGINPROVIDEROnGetUserInfo)(
        fwsDomain,
        FS_LPVOID clientData,
        FR_Login_UserInfo* pUserInfo
    );

} FR_CloudLoginProviderCallbacksRec,
*FR_CloudLoginProviderCallbacks;
```

#### **IStructSize**

The size of data structure. It must be set to *sizeof* ([FR\\_CloudLoginProviderCallbacksRec](#)).

#### **clientData**

The user-supplied data.

#### **wsOutUrl**

The user-supplied data.

#### **wsOutUrl)**

The user-supplied data.

## FR\_CPDFPluginProviderCallbacksRec

### Syntax

```
typedef struct _fr_cpdfpluginprovider_callbacks_ {
    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_BOOL (*OnShowConnectedPDFAdDialog)(FS_LPVOID clientData);
    void (*DoAfterApplySettings)(FS_LPVOID clientData);
    void (*DoConvertTocPDF)(FS_LPVOID clientData, FR_Document frDoc);
    void (*DoConvertTocPDF2)(FS_LPVOID clientData, FR_Document frDoc,
        FS_INT32 nOperType);
    void (*OpencPDFWebPage)(FS_LPVOID clientData, FS_BytString bsURL,
        FS_BytString bsTitle);
    void (*GetcPDFID)(FS_LPVOID clientData, FR_Document frDoc, FS_INT32
        nIdType, FS_BytString bsOutID);
    FS_BOOL (*IsJoinConnectedPDF)(FS_LPVOID clientData);
    void (*GetDocEndpoint)(FS_LPVOID clientData, FR_Document frdoc,
        FS_BytString bsOutEndpoint);
    void (*GetPDFDocEndpoint)(FS_LPVOID clientData, FPD_Document pdfdoc,
        FS_BytString bsOutEndpoint);
    FS_BOOL (*IsConnectedPDFDoc)(FS_LPVOID clientData, FR_Document
        frDoc);
    FS_BOOL(*SaveAsNewcPDF)(FS_LPVOID clientData, FR_Document frDoc,
        FS_WideString wsNewName, FS_WideString outNewPath);
} FR_CPDFPluginProviderCallbacksRec,
*FR_CPDFPluginProviderCallbacks;
```

#### **IStructSize**

The size of data structure. It must be set to *sizeof* ([FR\\_CPDFPluginProviderCallbacksRec](#)).

#### **clientData**

The user-supplied data.

#### **clientData)**

The user-supplied data.

#### **clientData)**

The user-supplied data.

#### **frDoc)**

The user-supplied data.

#### **nOperType)**

The user-supplied data.

#### **bsTitle)**

The user-supplied data.

**bsOutID)**

The user-supplied data.

**clientData)**

The user-supplied data.

**bsOutEndpoint)**

The user-supplied data.

**bsOutEndpoint)**

The user-supplied data.

**frDoc)**

The user-supplied data.

**outNewPath)**

The user-supplied data.

## FR>EditDrawNotifyCallbacksRec

### Syntax

```
typedef struct _fr_editdrawnotify_callbacks_{
    unsigned long lStructSize;
    FS_LVOID clientData;
    FS_BOOL (*FREditDrawNotifyGetData)(FS_LVOID clientData,
                                         FPD_RenderContext *pContext, FPD_RenderOptions* pOptions, void* pKey);
} FR>EditDrawNotifyCallbacksRec,
*FR>EditDrawNotifyCallbacks;
```

**lStructSize**

The size of data structure. It must be set to *sizeof( FR>EditDrawNotifyCallbacksRec )*.

**clientData**

The user-supplied data.

**pKey)**

The user-supplied data.

## FR>EditNotifyCallbacksRec

## Syntax

```
typedef struct _fr_editnotify_callbacks_ {
    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_FLOAT fSmallStep, FS_FLOAT fBigStep;
    FS_FLOAT fSmallStep, FS_FLOAT fBigStep;
    void (*FREditNotifyOnSetScrollPosX)(FS_LPVOID clientData, FS_FLOAT
    fx);
    void (*FREditNotifyOnSetScrollPosY)(FS_LPVOID clientData, FS_FLOAT
    fy);
    FS_FLOAT footX, FS_FLOAT footY, FR_VTWordPlace wordPlace);
    void (*FREditNotifyOnCaretChange)(FS_LPVOID clientData,
    FR_VTSecProps secProps, FR_VTWordProps wordProps);
    void (*FREditNotifyOnContentChange)(FS_LPVOID clientData, FS_FloatRect
    rcContent);
    void (*FREditNotifyOnInvalidateRect)(FS_LPVOID clientData, FS_FloatRect
    rect);
} FR>EditNotifyCallbacksRec, *FR>EditNotifyCallbacks;
```

### **IStructSize**

The size of data structure. It must be set to *sizeof*( [FR>EditNotifyCallbacksRec](#) ).

### **clientData**

The user-supplied data.

### **fBigStep)**

The user-supplied data.

### **fBigStep)**

The user-supplied data.

### **fx)**

The user-supplied data.

### **fy)**

The user-supplied data.

### **wordPlace)**

The user-supplied data.

### **wordProps)**

The user-supplied data.

### **rcContent)**

The user-supplied data.

### **rect)**

The user-supplied data.

## FR\_EditOprNotifyCallbacksRec

### Syntax

```
typedef struct _fr_editoprnotify_callbacks_ {
    unsigned long IStructSize;
    FS_LPVOID clientData;
    void (*FREditOprNotifyOnInsertWord)(FS_LPVOID clientData,
FR_VTWordPlace wordPlace, FR_VTWordPlace oldWordPlace);
    void (*FREditOprNotifyOnInsertReturn)(FS_LPVOID clientData,
FR_VTWordPlace wordPlace, FR_VTWordPlace oldWordPlace);
    void (*FREditOprNotifyOnBackSpace)(FS_LPVOID clientData,
FR_VTWordPlace wordPlace, FR_VTWordPlace oldWordPlace);
    void (*FREditOprNotifyonDelete)(FS_LPVOID clientData, FR_VTWordPlace
wordPlace, FR_VTWordPlace oldWordPlace);
    void (*FREditOprNotifyOnClear)(FS_LPVOID clientData, FR_VTWordPlace
wordPlace, FR_VTWordPlace oldWordPlace);
    void (*FREditOprNotifyOnInsertText)(FS_LPVOID clientData,
FR_VTWordPlace wordPlace, FR_VTWordPlace oldWordPlace);
    void (*FREditOprNotifyOnSetText)(FS_LPVOID clientData,
FR_VTWordPlace wordPlace, FR_VTWordPlace oldWordPlace);
    void (*FREditOprNotifyOnAddUndo)(FS_LPVOID clientData,
FR_Edit_UndoItem undolItem);
} FR>EditOprNotifyCallbacksRec, *FR>EditOprNotifyCallbacks;
```

#### **IStructSize**

The size of data structure. It must be set to *sizeof( FR\_EditOprNotifyCallbacksRec )*.

#### **clientData**

The user-supplied data.

#### **oldWordPlace)**

The user-supplied data.

**oldWordPlace)**

The user-supplied data.

**undoItem)**

The user-supplied data.

## FR\_EditUndoItemCallbacksRec

### Syntax

```
typedef struct _fr_editundoitem_callbacks_ {
    unsigned long IStructSize;
    FS_LPVVOID clientData;
    void (*FREditUndoItemUndo)(FS_LPVVOID clientData);
    void (*FREditUndoItemRedo)(FS_LPVVOID clientData);
    void (*FREditUndoItemGetUndoTitle)(FS_LPVVOID clientData,
FS_WideString outTitle);
    void (*FREditUndoItemRefersh)(FS_LPVVOID clientData);
    FS_BOOL (*FREditUndoItemIsPaint)(FS_LPVVOID clientData);
    void (*FREditUndoItemSetFirst)(FS_LPVVOID clientData, FS_BOOL bFirst);
    FS_BOOL (*FREditUndoItemIsFirst)(FS_LPVVOID clientData);
    void (*FREditUndoItemSetLast)(FS_LPVVOID clientData, FS_BOOL bLast);
    FS_BOOL (*FREditUndoItemIsLast)(FS_LPVVOID clientData);
    void (*FREditUndoItemRelease)(FS_LPVVOID clientData);
} FR>EditUndoItemCallbacksRec, *FR>EditUndoItemCallbacks;
```

**IStructSize**

The size of data structure. It must be set to *sizeof( FR>EditUndoItemCallbacksRec )*.

**clientData**

The user-supplied data.

**clientData)**

The user-supplied data.

**clientData)**

The user-supplied data.

**outTitle)**

The user-supplied data.

**clientData)**

The user-supplied data.

**clientData)**

The user-supplied data.

**bFirst)**

The user-supplied data.

**clientData)**

The user-supplied data.

**bLast)**

The user-supplied data.

**clientData)**

The user-supplied data.

**clientData)**

The user-supplied data.

## FR\_VariableTextProviderCallbacksRec

Variable text provider

### Syntax

```
typedef struct _fr_variabletextprovider_callbacks_ {
    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_INT32 (*FRVariableTextProviderGetCharWidth)(FS_LPVOID clientData,
        FS_INT32 nFontIndex, FS_WORD word, FS_INT32 nWordStyle);
    FS_INT32 (*FRVariableTextProviderGetCharItalicWidth)(FS_LPVOID
        clientData, FS_INT32 nFontIndex, FS_WORD word);
    FS_INT32 (*FRVariableTextProviderGetTypeAscent)(FS_LPVOID clientData,
        FS_INT32 nFontIndex);
    FS_INT32 (*FRVariableTextProviderGetTypeDescent)(FS_LPVOID
        clientData, FS_INT32 nFontIndex);
    FS_INT32 (*FRVariableTextProviderGetWordFontIndex)(FS_LPVOID
        clientData, FS_WORD word, FS_INT32 nFontIndex, FS_DWORD
        fontstyle, FS_DWORD nCharSet, FS_BOOL IsVertical);
    FS_BOOL (*FRVariableTextProviderIsLatinWord)(FS_LPVOID clientData,
        FS_WORD word);
    FS_INT32 (*FRVariableTextProviderGetDefaultFontIndex)(FS_LPVOID
        clientData);
    void (*FRVariableTextProviderSetDefaultFontIndex)(FS_LPVOID clientData,
        FS_INT32 nDefaultIndex);
    FS_BOOL (*FRVariableTextProviderIsVerticalFont)(FS_LPVOID clientData,
        FS_INT32 nFontindex);
    FR_Edit_FontMap (*FRVariableTextProviderGetFontMap)(FS_LPVOID
        clientData);
} FR_VariableTextProviderCallbacksRec,
*FR_VariableTextProviderCallbacks;
```

**IStructSize**

The size of data structure. It must be set to *sizeof*( [FR\\_VariableTextProviderCallbacksRec](#) ).

**clientData**

The user-supplied data.

**nWordStyle)**

The user-supplied data.

**word)**

The user-supplied data.

**nFontIndex)**

The user-supplied data.

**nFontIndex)**

The user-supplied data.

**IsVertical)**

The user-supplied data.

**word)**

The user-supplied data.

**clientData)**

The user-supplied data.

**nDefaultIndex)**

The user-supplied data.

**nFontindex)**

The user-supplied data.

**clientData)**

The user-supplied data.

## FR\_EmptyFrameWndViewEventHandlerCallbacksRec

A callback set for the event handler of the empty frame window view. See [FRAppCreateEmptyFrameViewEventHandler](#) .

### Syntax

```
typedef struct _fr_emptyframewndvieweventhandler_callbacks_{
    unsigned long lStructSize;
    FS_LPVVOID clientData;
    void (*FREmptyFrameViewOnActivate)(
        FS_LPVVOID clientData,
```

```
    HWND hView  
);  
  
void (*FREmptyFrameViewOnDeactivate)(  
    FS_LPVOID clientData,  
    HWND hView  
);  
  
void (*FREmptyFrameViewOnViewSize)(  
    FS_LPVOID clientData,  
    HWND hView,  
    FS_Rect rtClient  
);  
  
void (*FREmptyFrameViewOnSetFocus)(  
    FS_LPVOID clientData,  
    HWND hView,  
    HWND hOldWnd  
);  
  
void (*FREmptyFrameViewOnWillClose)(  
    FS_LPVOID clientData,  
    HWND hView  
);  
  
void (*FREmptyFrameViewOnViewCreate)(  
    FS_LPVOID clientData,  
    HWND hView  
);  
  
void (*FREmptyFrameViewOnDidClose)(  
    FS_LPVOID clientData,  
    HWND hFrameWnd  
);  
  
void (*FREmptyFrameViewOnWillDetachChildFrame)(  
    FS_LPVOID clientData,  
    HWND hFrameWnd  
);  
  
void (*FREmptyFrameViewOnActivate2)(  
    FS_LPVOID clientData,  
    HWND hView,  
    HWND hMainframe  
);  
  
void (*FREmptyFrameViewOnDeactivate2)(  
    FS_LPVOID clientData,  
    HWND hView,  
    HWND hMainframe  
);  
  
HWND (*FREmptyFrameViewGetDisplayViewHWnd)(  
    FS_LPVOID clientData,  
    HWND hPluginView  
);
```

---

```
 } FR_EmptyFrameWndViewEventHandlerCallbacksRec,
 *FR_EmptyFrameWndViewEventHandlerCallbacks;
```

**IStructSize**

The size of data structure. It must be set to *sizeof* ([FR\\_EmptyFrameWndViewEventHandlerCallbacksRec](#)).

**clientData**

The user-supplied data.

**FR\_FormatToolCallbacksRec**

The callback set for format tool. The callbacks are called by Foxit Reader when the events occur. You can set the callback set to format tool through [FRFormatToolsSetEvent](#).

**Syntax**

```
typedef struct _fr_formattool_callbacks_{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    void (*FRFormatToolOnFontNameChanged)(
        FS_LPVOID clientData,
        FS_LPCWSTR lpwsFontName,
        FS_LPCWSTR lpwsScriptName
    );
    void (*FRFormatToolOnFontSizeChanged)(
        FS_LPVOID clientData,
        FS_FLOAT flFontSize
    );
    void (*FRFormatToolOnTextColorChanged)(
        FS_LPVOID clientData,
        COLORREF textColor
    );
    void (*FRFormatToolOnLineColorChanged)(
        FS_LPVOID clientData,
        COLORREF lineColor,
        FS_FLOAT bTransparent
    );
    void (*FRFormatToolOnFillColorChanged)(
        FS_LPVOID clientData,
        COLORREF FillColor,
        FS_FLOAT bTransparent
    );
    void (*FRFormatToolOnBoldChanged)(
        FS_LPVOID clientData,
        FS_FLOAT bBold,
```

```
    FS_FLOAT bEnabled
);

void (*FRFormatToolOnItalicChanged)(
    FS_LPVOID clientData,
    FS_FLOAT bItalic,
    FS_FLOAT bEnabled
);

void (*FRFormatToolOnAlignChanged)(
    FS_LPVOID clientData,
    FS_DWORD dwAlign
);

void (*FRFormatToolOnCharSpaceChanged)(
    FS_LPVOID clientData,
    FS_FLOAT flSpace
);

void (*FRFormatToolOnCharHorzScaleChanged)(
    FS_LPVOID clientData,
    FS_INT32 nScale
);

void (*FRFormatToolOnLineLeadingChanged)(
    FS_LPVOID clientData,
    FS_FLOAT flLineLeading
);

void (*FRFormatToolOnUnderlineChanged)(
    FS_LPVOID clientData,
    FS_FLOAT bUnderline
);

void (*FRFormatToolOnCrossChanged)(
    FS_LPVOID clientData,
    FS_FLOAT bCross
);

void (*FRFormatToolOnSuperScriptChanged)(
    FS_LPVOID clientData,
    FS_FLOAT bSuperSet
);

void (*FRFormatToolOnSubScriptChanged)(
    FS_LPVOID clientData,
    FS_FLOAT bSubSet
);

void (*FRFormatToolOnInDentChanged)(
    FS_LPVOID clientData
);

void (*FRFormatToolOnDeDentChanged)(
    FS_LPVOID clientData
);

void (*FRFormatToolOnWordSpaceChanged)(
```

---

```

    FS_LPVOID clientData,
    FS_FLOAT fWordSpace
);

void (*FRFormatToolOnArrangeAlign)(
    FS_LPVOID clientData,
    FRFormatToolArrangeAlignInfo nAlign
);

void (*FRFormatToolOnArrangeCenter)(
    FS_LPVOID clientData,
    FRFormatToolArrangeCenterInfo nCenter
);

void (*FRFormatToolOnArrangeDistribute)(
    FS_LPVOID clientData,
    FRFormatToolArrangeDistributeInfo nDistribute
);

void (*FRFormatToolGetArrangeEnable)(
    FS_LPVOID clientData,
    FRFormatToolArrangeInfo nInfo,
    FS_BOOL* outEnable
);

void (*FRFormatToolGetLineColorEnableNoColor)(
    FS_LPVOID clientData,
    FS_BOOL* outEnable
);

void (*FRFormatToolGetFillColorEnableNoColor)(
    FS_LPVOID clientData,
    FS_BOOL* outEnable
);

void (*FRFormatToolOnWritingDirChanged)(
    FS_LPVOID clientData,
    FRFormatToolWritingDirection eDir
);

void (*FRFormatToolOnOpacityChanged)(
    FS_LPVOID clientData,
    FS_INT32 nOpacity
);

void (*FRFormatToolOnOpacityScroll)(
    FS_LPVOID clientData,
    FS_INT32 nOpacity
);
}

FR_FormatToolCallbacksRec, *FR_FormatToolCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FR\_FormatToolCallbacksRec )*.

**clientData**

The user-supplied data.

## FR\_FoxitBrowserEventCallbacksRec

A callback set for Foxit browser event handler. These callbacks will be invoked when some events of the Foxit browser occurs. See [FRHTMLMgrRegisterFoxitBrowserEventHandler](#) .

### Syntax

```
typedef struct _fr_foxitbrowserevent_callbacks_{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    void (*FRFoxitBrowserEventRelease)(FS_LPVOID clientData
    );
    void (*FRFoxitBrowserEventOnDelFavLinks)(FS_LPVOID clientData,
                                              FS_LPCWSTR lpwsUrl
                                              );
} FR_FoxitBrowserEventCallbacksRec,
*FR_FoxitBrowserEventCallbacks;
```

#### **IStructSize**

The size of data structure. It must be set to *sizeof( FR\_FoxitBrowserEventCallbacksRec )*.

#### **clientData**

The user-supplied data.

## FR\_HMLEventCallbacksRec

A callback set for HTML window event handler. These callbacks will be invoked when the HTML window is created, destroyed and so on. See [FRHTMLMgrRegisterHTMLEventHandler](#) .

### Syntax

```
typedef struct _fr_htmlevent_callbacks_{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    void (*FRHTMLEventRelease)(FS_LPVOID clientData
    );
    void (*FRHTMLEventOnDocFrameCreate)(FS_LPVOID clientData,
```

---

```

HWND hFrameWnd,
HWND hView,
FS_BOOL bNeedCreatePanel
);

FS_BOOL (*FRHTMLEventOnDocViewBeforeNavigate2)(
    FS_LPVOID clientData,
    HWND hView,
    FS_LPCWSTR lpURL
);

void (*FRHTMLEventOnDocViewBeforeDestory)(
    FS_LPVOID clientData,
    HWND hView,
    FS_LPCWSTR lpURL
);

FS_BOOL (*FRHTMLEventOnDocViewNewWindow3)(
    FS_LPVOID clientData,
    void ppDisp,
    FS_BOOL Cancel,
    FS_DWORD dwFlags,
    FS_LPCWSTR bstrUrlContext,
    FS_LPCWSTR bstrUrl
);

FS_BOOL (*FRHTMLEventOnWillFrameClose)(
    FS_LPVOID clientData
);

FS_BOOL (*FRHTMLEventOnDispatchFun)(
    FS_LPVOID clientData,
    HWND hView,
    FS_LPCWSTR lpwsModule,
    FS_LPCWSTR lpwsFunName,
    FS_LPCWSTR lpwsParam,
    FS_WideString outRet
);
}

} FR_HTMLEventCallbacksRec, *FR_HTMLEventCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( [FR\\_HTMLEventCallbacksRec](#) )*.

**clientData**

The user-supplied data.

**FR\_MenuOwnerDrawCallbacksRec**

A callback set. It is called by foxit reader when it is drawing the menu. See [FRMenuRegisterOwnerDrawHandle](#)

## Syntax

```
typedef struct _fr_menu_callbacks_ {
    unsigned long IStructSize;
    FS_BOOL (*FRMenuItemOnMeasureItem)(  

        FR_MenuItem menuItem,  

        FS_INT32 width,  

        FS_INT32 height  

    );
    FS_BOOL (*FRMenuItemOnDrawItem)(  

        FR_MenuItem menuItem,  

        HDC hDC,  

        const FS_Rect rect  

    );
    void (*FRMenuItemOnInitMenuPopup)(  

        FR_Menu menu  

    );
} FR_MenuOwnerDrawCallbacksRec,  

*FR_MenuOwnerDrawCallbacks;
```

### IStructSize

The size of data structure. It must be set to *sizeof* ([FR\\_MenuOwnerDrawCallbacksRec](#)).

## FR\_PanelEventCallbacksRec

A callback set for the panel event handler. These callbacks will be invoked when an event occurs like the panel size changing.

## Syntax

```
typedef struct _fr_panelevent_callbacks_ {
    unsigned long IStructSize;
    FS_LPVVOID clientData;
    FS_BOOL (*FRPanelEventOnShowPanelMenu)(  

        FS_LPVVOID clientData,  

        HWND hParent,  

        FS_INT32 posX,  

        FS_INT32 posY  

    );
    void (*FRPanelEventOnPanelActive)(  

        FS_LPVVOID clientData,  

        FS_LPCSTR lpsName  

    );
    void (*FRPanelEventOnPanelSize)(  

        FS_LPVVOID clientData,  

        FS_LPCSTR lpsName,  

        FS_Rect rect,  

    );
```

```

HWND hChildFrm
);

void (*FRPanelEventOnMouseMove)(  

    FS_LPVOID clientData,  

    FS_INT32 x,  

    FS_INT32 y  

);
}

} FR_PanelEventCallbacksRec, *FR_PanelEventCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FR\_PanelEventCallbacksRec )*.

**clientData**

The user-supplied data.

**FR\_PropertyToolCallbacksRec**

The callback set for property tool.

**Syntax**

```

typedef struct _fr_propertytool_callbacks_ {  

    unsigned long IStructSize;  

    FS_LPVOID clientData;  

    void (*FRPropertyToolOnColorChanged)(  

        FS_LPVOID clientData,  

        COLORREF color  

    );  

  

    void (*FRPropertyToolOnOpacityChanged)(  

        FS_LPVOID clientData,  

        FS_INT32 nOpacity  

    );  

  

    void (*FRPropertyToolOnBeginScroll)(  

        FS_LPVOID clientData  

    );  

  

    void (*FRPropertyToolOnEndScroll)(  

        FS_LPVOID clientData  

    );  

  

    void (*FRPropertyToolOnOpacityScroll)(  

        FS_LPVOID clientData,  

        FS_INT32 nOpacity  

    );  

}

} FR_PropertyToolCallbacksRec, *FR_PropertyToolCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof*( [FR\\_PropertyToolCallbacksRec](#) ).

**clientData**

The user-supplied data.

## **FR\_ResourcePropertyNotifyCallbacksRec**

The callback set for property box. The callbacks are called by Foxit Reader when the events occur.

### Syntax

```
typedef struct _fr_resourceproperty_notify_callbacks_{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    void (*\_FRResourcePropertyNotifyOnLockObjects)(FS_LPVOID clientData,
                                                FS_BOOL bLocked
                                              );
}

} FR_ResourcePropertyNotifyCallbacksRec,
*FR_ResourcePropertyNotifyCallbacks;
```

**IStructSize**

The size of data structure. It must be set to *sizeof*( [FR\\_ResourcePropertyNotifyCallbacksRec](#) ).

**clientData**

The user-supplied data.

## **FR\_ResourcePropertyPageCallbacksRec**

The callback set for property page. The callbacks are called by Foxit Reader to add new property page to the property box. You can register the callbacks through [FRResourcePropertyBoxRegisterPropertyPage](#) .

### Syntax

```
typedef struct _fr_resourceproperty_page_callbacks_{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    void (*\_FRResourcePropertyPageSetPropertyBox)(FS_LPVOID clientData,
                                                FR\_ResourcePropertyBox pPropertyBox
                                              );
}
```

```

void (*FRResourcePropertyPageGetPageTitle)(
    FS\_LPVOID clientData,
    FS\_WideString outTitle
);

void (*FRResourcePropertyPageGetPageName)(
    FS\_LPVOID clientData,
    FS\_WideString outName
);

FS\_INT32 (*FRResourcePropertyPageGetSequenceNumber)(
    FS\_LPVOID clientData
);

FS\_BOOL (*FRResourcePropertyPageIsSourceSupported)(
    FS\_LPVOID clientData,
    FS\_INT32 nSource,
    FR\_ResourcePropertySource pSourceFunc
);

void (*FRResourcePropertyPageUpdatePage)(
    FS\_LPVOID clientData
);

HWND (*FRResourcePropertyPageGetPageHWnd)(
    FS\_LPVOID clientData
);

FS\_BOOL (*FRResourcePropertyPageGetPageSize)(
    FS\_LPVOID clientData,
    FS\_INT32* outCX,
    FS\_INT32* outCY
);

FS\_BOOL (*FRResourcePropertyPageCreatePage)(
    FS\_LPVOID clientData,
    HWND hParent
);

FS\_BOOL (*FRResourcePropertyPageOnDeactive)(
    FS\_LPVOID clientData
);

void (*FRResourcePropertyPageOnLangUIChange)(
    FS\_LPVOID clientData
);

} FR_ResourcePropertyPageCallbacksRec,
*FR_ResourcePropertyPageCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( [FR\\_ResourcePropertyPageCallbacksRec](#) )*.

**clientData**

The user-supplied data.

## FR\_ResourcePropertySourceCallbacksRec

The callback set for property source. The callbacks are called by Foxit Reader to deal with the source of the property box. See [FRResourcePropertyBoxRegisterSourceType](#) .

### Syntax

```
typedef struct _fr_resourceproperty_source_callbacks_{
    unsigned long lStructSize;
    FS_LPVVOID clientData;
    FR_Document (*FRResourcePropertySourceGetFRDocument)(
        FS_LPVVOID clientData
    );
    FS_INT32 (*FRResourcePropertySourceCountSelectObjects)(
        FS_LPVVOID clientData
    );
    void* (*FRResourcePropertySourceGetSelectObject)(
        FS_LPVVOID clientData,
        FS_INT32 index
    );
    FS_INT32 (*FRResourcePropertySourceGetSelectObjectType)(
        FS_LPVVOID clientData,
        FS_ByteStringArray outArray
    );
    FS_BOOL (*FRResourcePropertySourceIsLockedObjectExisted)(
        FS_LPVVOID clientData
    );
    FS_BOOL (*FRResourcePropertySourceIsDisabledObjectExisted)(
        FS_LPVVOID clientData
    );
    FS_BOOL (*FRResourcePropertySourceIsPropertyBoxEnabled)(
        FS_LPVVOID clientData,
        FS_LPCSTR lpsObjectType
    );
    void (*FRResourcePropertySourceGetPropertyBoxTitle)(
        FS_LPVVOID clientData,
        FS_LPCSTR lpsObjectType,
        FS_WideString outTitle
    );
    FS_BOOL (*FRResourcePropertySourceCanExistedSetDefaultButton)(
        FS_LPVVOID clientData
    );
    void (*FRResourcePropertySourceOnBoxClose)()
```

```

FS_LPVOID clientData,
FS_INT32 nType,
FS_INT32 bSetDefaultButton
);

} FR_ResourcePropertySourceCallbacksRec,
*FR_ResourcePropertySourceCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof* ([FR\\_ResourcePropertySourceCallbacksRec](#)).

**clientData**

The user-supplied data.

**FR\_RibbonFilePageEventCallbacksRec**

A callback set for ribbon file page event handler. See [FRRibbonBarRegisterFilePageEventHandler](#).

**Syntax**

```

typedef struct _fr_ribbonfilepageevent_callbacks_ {
    unsigned long IStructSize;
    FS_LPVOID clientData;
    void (*FROnRemovePropertyPage\)\(
        FS\_LPVOID clientData,
        FS\_LPCSTR lpsName
    \);

    void \(\*FROnClickAddPlacePageButton\\)\\(
        FS\\_LPVOID clientData,
        FS\\_LPCSTR lpsButtonName,
        FS\\_BOOL bSaveAsItemPage
    \\);

    void \\(\\*FROnSelectPropertyPage\\\)\\\(
        FS\\\_LPVOID clientData,
        FS\\\_LPCSTR lpsElementName,
        FS\\\_LPCSTR lpsPageName
    \\\);
}

} FR\\\_RibbonFilePageEventCallbacksRec,
\\\*FR\\\_RibbonFilePageEventCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof* ([FR\\_RibbonFilePageEventCallbacksRec](#)).

**clientData**

The user-supplied data.

## FR\_TabBandAddBtnCallbacksRec

A callback set for adding button to the tab band.

### Syntax

```
typedef struct _fr_tabbandaddbtn_callbacks_ {
    unsigned long IStructSize;
    FS_LPVOID clientData;
    void (*FRTabBandOnClickBtn)(
        FS_LPVOID clientData
    );

    void (*FRTabBandGetTooltip)(
        FS_LPVOID clientData,
        FS_WideString outTooltip
    );

    FS_BOOL (*FRTabBandTopPriority)(
        FS_LPVOID clientData
    );
} FR_TabBandAddBtnCallbacksRec,
*FR_TabBandAddBtnCallbacks;
```

#### IStructSize

The size of data structure. It must be set to *sizeof( FR\_TabBandAddBtnCallbacksRec )*.

#### clientData

The user-supplied data.

## FR\_CmdMsgEventCallbacksRec

A callback set for application-level event handler. It is called by the *Foxit Reader* to route and dispatch command messages and to handle the update of command user-interface objects, such as menu, toolbar. See [FRAppRegisterCmdMsgEventHandler](#) .

### Syntax

```
typedef struct _fr_cmcmdsgevent_callbacks_ {
    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_BOOL (*FRCmdMsgOnCmdMsgByName)(
        FS_LPVOID clientData,
        FS_LPCSTR lpsName,
        FS_INT32 nCode,
```

```

        void* pExtra,
        void* pHandlerInfo
    );
}

} FR_CmdMsgEventCallbacksRec, *FR_CmdMsgEventCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof*( [FR\\_CmdMsgEventCallbacksRec](#) ).

**clientData**

The user-supplied data.

**FR\_ToolCallbacksRec**

A callback set for [FR\\_Tool](#) . You can use the callback set to control a tool. It can also receive the message of event, such as [FRTToolOnKeyDown](#) , [FRTToolOnLButtonDown](#) and so on. See [FRTToolNew](#) .

**Syntax**

```

typedef struct _fr_tool_callbacks_{
    unsigned long IStructSize;
    FS\_LPVOID clientData;
    FS\_BOOL (*FRTToolOnInit)(
        FS\_LPVOID clientData
    );

    FS\_BOOL (*FRTToolDestroy)(
        FS\_LPVOID clientData
    );

    void (*FRTToolOnActivate)(
        FS\_LPVOID clientData,
        FS\_BOOL bPersistent
    );

    void (*FRTToolOnDeactivate)(
        FS\_LPVOID clientData
    );

    FS\_BOOL (*FRTToolOnKeyDown)(
        FS\_LPVOID clientData,
        FS\_UINT nKeyCode,
        FS\_UINT nFlags
    );

    FS\_BOOL (*FRTToolOnKeyUp)(
        FS\_LPVOID clientData,
        FS\_UINT nKeyCode,
        FS\_UINT nFlags
    );
}

```

```
FS_BOOL (*FRToolOnChar)(  
    FS_LPVOID clientData,  
    FS_UINT nChar,  
    FS_UINT nFlags  
);  
  
void (*FRToolOnLeavePage)(  
    FS_LPVOID clientData,  
    FR_PageView pageview  
);  
  
FS_BOOL (*FRToolIsEnabled)(  
    FS_LPVOID clientData  
);  
  
FS_BOOL (*FRToolOnLButtonDown)(  
    FS_LPVOID clientData,  
    FR_PageView pageview,  
    FS_UINT nFlags,  
    FS_DevicePoint point  
);  
  
FS_BOOL (*FRToolOnLButtonUp)(  
    FS_LPVOID clientData,  
    FR_PageView pageview,  
    FS_UINT nFlags,  
    FS_DevicePoint point  
);  
  
FS_BOOL (*FRToolOnLButtonDblClk)(  
    FS_LPVOID clientData,  
    FR_PageView pageview,  
    FS_UINT nFlags,  
    FS_DevicePoint point  
);  
  
FS_BOOL (*FRToolOnMouseMove)(  
    FS_LPVOID clientData,  
    FR_PageView pageview,  
    FS_UINT nFlags,  
    FS_DevicePoint point  
);  
  
FS_BOOL (*FRToolOnRButtonDown)(  
    FS_LPVOID clientData,  
    FR_PageView pageview,  
    FS_UINT nFlags,  
    FS_DevicePoint point  
);  
  
FS_BOOL (*FRToolOnRButtonUp)(  
    FS_LPVOID clientData,  
    FR_PageView pageview,  
    FS_UINT nFlags,  
    FS_DevicePoint point  
);  
  
FS_BOOL (*FRToolOnRButtonDblClk)(
```

---

```

FS_LPVOID clientData,
FR_PageView pageview,
FS_UINT nFlags,
FS_DevicePoint point
);

FS_BOOL (*FRTToolOnMouseWheel)(
    FS_LPVOID clientData,
    FR_PageView pageview,
    FS_UINT nFlags,
    FS_SHORT zDelta,
    FS_DevicePoint point
);

FS_BOOL (*FRTToolOnDraw)(
    FS_LPVOID clientData,
    FR_WinPort winPort,
    FS_DWORD dwFlags
);

FS_BOOL (*FRTToolIsProcessing)(
    FS_LPVOID clientData
);

FS_BOOL (*FRTToolOnMouseHover)(
    FS_LPVOID clientData,
    FR_PageView pageview,
    FS_DevicePoint point
);

FS_BOOL (*FRTToolIsWndCapturing)(
    FS_LPVOID clientData
);

} FR_ToolCallbacksRec, *FR_ToolCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FR\_ToolCallbacksRec )*.

**clientData**

The user-supplied data.

**FRMS\_CommonEventCallbacksRec**

A callback set for RMS common event handler.

**Syntax**

```

typedef struct _frms_commonevent_callbacks_ {
    unsigned long IStructSize;
    void* clientData;

```

```

void (*FRMSCommonOnDidInitSecureEnvironment)(FS_LPVOID
clientData);
void (*FRMSCommonOnWillGetDefServer)(FS_LPVOID clientData);
void (*FRMSCommonOnDidGetDefServer)(FS_LPVOID clientData,
FS_LPWSTR pwszCertURL, FS_LPWSTR pwszLicensingURL);
void (*FRMSCommonOnWillGetDefCredential)(FS_LPVOID clientData);
void (*FRMSCommonOnDidGetDefCredential)(FS_LPVOID clientData,
FS_LPWSTR pwszUserEmail);
void (*FRMSCommonOnWillActivateMachine)(FS_LPVOID clientData);
void (*FRMSCommonOnDidActivateMachine)(FS_LPVOID clientData,
FS_LPWSTR pwszMachineCert);
void (*FRMSCommonOnWillActivateCredential)(FS_LPVOID clientData,
FS_LPWSTR pwszUserEmail);
void (*FRMSCommonOnDidActivateCredential)(FS_LPVOID clientData,
FS_LPWSTR pwszUserEmail, FS_LPWSTR pwszRAC);
void (*FRMSCommonOnSelCredential)(FS_LPVOID clientData,
FS_LPWSTR pwszUserEmail);
void (*FRMSCommonOnWillInitSecureEnvironment)(
    FS_LPVOID clientData,
    FS_LPWSTR pwszMachineCert,
    FS_LPWSTR pwszManifest
);

} FRMS_CommonEventCallbacksRec,
*FRMS_CommonEventCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof*( [FRMS\\_CommonEventCallbacksRec](#) ).

**clientData**

The user-supplied data.

**clientData)**

The user-supplied data.

**clientData)**

The user-supplied data.

**pwszLicensingURL)**

The user-supplied data.

**clientData)**

The user-supplied data.

**pwszUserEmail)**

The user-supplied data.

**clientData)**

The user-supplied data.

**pwszMachineCert)**

The user-supplied data.

**pwszUserEmail)**

The user-supplied data.

**pwszRAC)**

The user-supplied data.

**pwszUserEmail)**

The user-supplied data.

## FRMS\_DecryptionCusAuthCallbacksRec

A callback set for RMS decryption custom authorization handler.

### Syntax

```
typedef struct _frms_decryptioncusauth_callbacks_ {
    unsigned long IStructSize;
    void* clientData;
    FS_BOOL (*FRMSDecryptionCusAuth)(FS_LPVOID clientData,
                                         FPD_Document doc,
                                         FS_LPWSTR pwszSIL,
                                         FS_LPWSTR pwszEULChain
                                         );
}

} FRMS_DecryptionCusAuthCallbacksRec,
*FRMS_DecryptionCusAuthCallbacks;
```

**IStructSize**

The size of data structure. It must be set to *sizeof( [FRMS\\_DecryptionCusAuthCallbacksRec](#) )*.

**clientData**

The user-supplied data.

## FRMS\_DecryptionEventCallbacksRec

A callback set for RMS Decryption event handler.

### Syntax

```
typedef struct _frms_decryptionevent_callbacks_ {
    unsigned long IStructSize;
    void* clientData;
```

```

void (*FRMSDecryptionOnWillAuthentication)(FS_LPVOID clientData,
FPD_Object encryptDict, FPD_Document doc);
void (*FRMSDecryptionOnDidAuthentication)(FS_LPVOID clientData,
FPD_Object encryptDict, FPD_Document doc, FS_LPWSTR pwszUserEmail);
void (*FRMSDecryptionOnWillGetEULFromLocal)(FS_LPVOID clientData,
FPD_Document doc, FS_LPWSTR pwszUserEmail);
void (*FRMSDecryptionOnDidGetEULFromLocal)(FS_LPVOID clientData,
FPD_Document doc, FS_LPWSTR pwszUserEmail, FS_LPWSTR
pwszEULChain);
void (*FRMSDecryptionOnWillAcquireEUL)(FS_LPVOID clientData,
FPD_Document doc, FS_LPWSTR pwszUserEmail);
void (*FRMSDecryptionOnDidAcquireEUL)(FS_LPVOID clientData,
FPD_Document doc, FS_LPWSTR pwszUserEmail, FS_LPWSTR
pwszEULChain);
void (*FRMSDecryptionOnWillCreateBoundLicense)(FS_LPVOID clientData,
FPD_Document doc, FS_LPWSTR pwszUserEmail, FS_LPWSTR
pwszEULChain, FS_LPWSTR pwszRight);
void (*FRMSDecryptionOnDidCreateBoundLicense)(FS_LPVOID clientData,
FPD_Document doc, FS_LPWSTR pwszUserEmail, FS_LPWSTR
pwszEULChain, FS_LPWSTR pwszRight);
void (*FRMSDecryptionOnDecryptStart)(FS_LPVOID clientData,
FPD_Document doc, FS_DWORD objnum, FS_DWORD gennum);
void (*FRMSDecryptionOnDecryptContent)(FS_LPVOID clientData,
FPD_Document doc, FS_LPCBYTE src_buf, FS_DWORD src_size);
void (*FRMSDecryptionOnReEncryptContent)(FS_LPVOID clientData,
FPD_Document doc, FS_DWORD objnum, FS_DWORD version,
FS_LPCBYTE src_buf, FS_DWORD src_size);
} FRMS_DecryptionEventCallbacksRec,
*FRMS_DecryptionEventCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FRMS\_DecryptionEventCallbacksRec )*.

**clientData**

The user-supplied data.

**doc)**

The user-supplied data.

**pwszUserEmail)**

The user-supplied data.

**pwszUserEmail)**

The user-supplied data.

**pwszEULChain)**

The user-supplied data.

**pwszUserEmail)**

The user-supplied data.

**pwszEULChain)**

The user-supplied data.

**pwszRight)**

The user-supplied data.

**pwszRight)**

The user-supplied data.

**gennum)**

The user-supplied data.

**src\_size)**

The user-supplied data.

**src\_size)**

The user-supplied data.

## FRMS\_EncryptionEventCallbacksRec

A callback set for RMS Encryption event handler.

### Syntax

```
typedef struct _frms_encryptionevent_callbacks_{
    unsigned long IStructSize;
    void* clientData;
    void (*FRMSEncryptionOnWillGetDefTemplate)(FS_LPVOID clientData);
    void (*FRMSEncryptionOnDidGetDefTemplate)(FS_LPVOID clientData,
        FS_LPWSTR pwszPath);
    void (*FRMSEncryptionOnWillAcquireCLC)(FS_LPVOID clientData,
        FS_LPWSTR pwszUserEmail);
    void (*FRMSEncryptionOnDidAcquireCLC)(FS_LPVOID clientData,
        FS_LPWSTR pwszUserEmail, FS_LPWSTR pwszCLC);
    void (*FRMSEncryptionOnWillGetSignedIL)(FS_LPVOID clientData,
        FPD_Document pdfDoc, FS_LPWSTR pwszCLC);
    void (*FRMSEncryptionOnDidGetSignedIL)(FS_LPVOID clientData,
        FPD_Document pdfDoc, FS_LPWSTR pwszSignedIL);
    void (*FRMSEncryptionOnGetOwnerLicense)(FS_LPVOID clientData,
        FS_LPWSTR pwszOwnerlicense);
    void (*FRMSEncryptionOnWillEncrypt)(FS_LPVOID clientData,
        FPD_Document pdfDoc);
    void (*FRMSEncryptionOnEncryptContent)(FS_LPVOID clientData,
        FPD_Document pdfDoc, FS_DWORD objnum, FS_DWORD version,
        FS_LPCBYTE src_buf, FS_DWORD src_size);
    void (*FRMSEncryptionOnDidEncrypt)(FS_LPVOID clientData,
        FPD_Document pdfDoc);
    void (*FRMSEncryptionOnSelTemplate)(FS_LPVOID clientData,
        FS_LPWSTR pwszPath);
} FRMS_EncryptionEventCallbacksRec,
*FRMS_EncryptionEventCallbacks;
```

**lStructSize**

The size of data structure. It must be set to *sizeof( FRMS\_EncryptionEventCallbacksRec )*.

**clientData**

The user-supplied data.

**pwszPath)**

The user-supplied data.

**pwszUserEmail)**

The user-supplied data.

**pwszCLC)**

The user-supplied data.

**pwszCLC)**

The user-supplied data.

**pwszSignedIL)**

The user-supplied data.

**pwszOwnerlicense)**

The user-supplied data.

**pdfDoc)**

The user-supplied data.

**src\_size)**

The user-supplied data.

**pdfDoc)**

The user-supplied data.

**pwszPath)**

The user-supplied data.

## FPD\_BackgroundDraw

PDF background drawing callback struct. See [FPD\\_BackgroundDrawHandler](#) .

## Syntax

---

```
typedef struct __FPD_BackgroundDraw__{
    unsigned long IStructSize;
    void (*OnDrawBackground)(FPD_RenderDevice bitmapDevice,
        FS_AffineMatrix original2Bitmap
    );
}

} FPD_BackgroundDraw, *PFPD_BackgroundDraw;
```

**IStructSize**

The size of data structure. It must be set to *sizeof( FPD\_BackgroundDraw )*.

**FPD\_CreatorOptionCallbacksRec**

A callback set for creator option. See [FPDCreatorSetOption](#) . The callbacks will be invoked when the PDF creator is decoding or encoding content progressively. The plug-in has to implement the callback function as needed.

**Syntax**

```
typedef struct _fpd_creatoroption_callbacks__{
    unsigned long IStructSize;
    FS_LPVOID clientData;
    FS_FileStream (*FPDCreatorOptionGetTempFile)(FS_LPVOID clientData,
        FPD_Object obj
    );

    void (*FPDCreatorOptionReleaseTempFile)(FS_LPVOID clientData,
        FS_FileStream fileStream
    );
}

} FPD_CreatorOptionCallbacksRec,
*FPD_CreatorOptionCallbacks;
```

**IStructSize**

The size of data structure. It must be set to *sizeof( FPD\_CreatorOptionCallbacksRec )*.

**clientData**

The user-supplied data.

**FPD\_EnumPage**

A callback for [FPD\\_EnumPageHandler](#) . See [FPDDocEnumPages](#) .

## Syntax

```
typedef struct {
    unsigned long IStructSize;
    FS_LPVVOID clientData;
    FS_BOOL (*EnumPage)(
        FS_LPVVOID clientData,
        FPD_Object pageDict
    );
}

} FPD_EnumPage, *PFPD_EnumPage;
```

### **IStructSize**

The size of data structure. It must be set to *sizeof( FPD\_EnumPage )*.

### **clientData**

The user-supplied data.

## FPD\_FormNotifyCallbacksRec

A set of callbacks for PDF form notify handler.

## Syntax

```
typedef struct _fpd_formnotify_callbacks_{
    unsigned long IStructSize;
    FS_LPVVOID clientData;
    FS_INT32 (*BeforeValueChange)(
        FS_LPVVOID clientData,
        FPD_FormField field,
        FS_WideString csValue
    );
    FS_INT32 (*AfterValueChange)(
        FS_LPVVOID clientData,
        FPD_FormField field
    );
    FS_INT32 (*BeforeSelectionChange)(
        FS_LPVVOID clientData,
        FPD_FormField field,
        FS_WideString csValue
    );
    FS_INT32 (*AfterSelectionChange)(
        FS_LPVVOID clientData,
        FPD_FormField field
    );
    FS_INT32 (*AfterCheckedStatusChange)(
        FS_LPVVOID clientData,
```

```

FPD_FormField field,
FS_ByteArray statusArray
);

FS_INT32 (*BeforeFormReset)(
    FS_LPVOID clientData,
    FPD_InterForm form
);

FS_INT32 (*AfterFormReset)(
    FS_LPVOID clientData,
    FPD_InterForm form
);

FS_INT32 (*BeforeFormImportData)(
    FS_LPVOID clientData,
    FPD_InterForm form
);

FS_INT32 (*AfterFormImportData)(
    FS_LPVOID clientData,
    FPD_InterForm form
);

} FPD_FormNotifyCallbacksRec, *FPD_FormNotifyCallbacks;

```

**IStructSize**

The size of data structure. It must be set to *sizeof( FPD\_FormNotifyCallbacksRec )*

**clientData**

The user-supplied data.

## FPD\_OCContextCallBack

The PDF optional content context callback struct. See [FPD\\_OCContextHandler](#) .

### Syntax

```

typedef struct __FPD_OCContextCallBack__{
    unsigned long IStructSize;
    FS_BOOL (*CheckOCGVisible)(
        FPD_Object ogc
    );

    FS_BOOL (*CheckObjectVisible)(
        FPD_PageObject obj
    );
}

} FPD_OCContextCallBack, *PFPD_OCContextCallBack;

```

**IStructSize**

The size of data structure. It must be set to *sizeof*( [FPD\\_OCCallback](#) ).

## FPD\_ProgressiveEncryptCallbacksRec

A callback set for progressive encrypt handler. See [FPDCreatorSetProgressiveEncryptHandler](#). The callbacks will be invoked when the PDF creator is encrypting content progressively. The plug-in has to implement the callback function as needed.

### Syntax

```
typedef struct _fpd_progressiveencrypt_callbacks_{
    unsigned long lStructSize;
    FS_LVOID clientData;
    FS_FileStream (*FPDProgressiveEncrypGetTempFile)(
        FS_LVOID clientData
    );

    void (*FPDProgressiveEncrypReleaseTempFile)(
        FS_LVOID clientData,
        FS_FileStream fileStream
    );
} FPD_ProgressiveEncryptCallbacksRec,
*FPD_ProgressiveEncryptCallbacks;
```

#### **lStructSize**

The size of data structure. It must be set to *sizeof*( [FPD\\_ProgressiveEncryptCallbacksRec](#) ).

#### **clientData**

The user-supplied data.